

# Interactive Theorem Proving

## Lecture 1

Cezary Kaliszyk

March 8, 2013



# Administration

## Teacher

- Cezary Kaliszyk
- Consultation hours: Thursday afternoon, 3M12

## Grading

- Lecture: Closed book exam
- Exercises: Assignments and participation

## Practical Assignments

- Software / Logins

# Course content

## Foundations

- Minimal Logic and Natural Deduction
- Lambda Calculus with Extensions

## LCF style

- HOL Light

## Proof Assistants

- Type Theory and Coq
- Set Theory and Mizar
- Logical Frameworks

# What is a Proof Assistant? (1/2)

## A Proof Assistant is a

- a computer program
- to assist a mathematician
- in the production of a proof
- that is mechanically checked

## What does a Proof Assistant do?

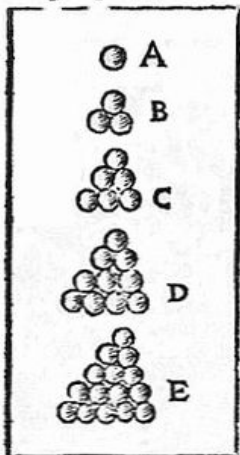
- Keep track of theories, definitions, assumptions
- Interaction - proof editing
- Proof checking
- Automation - proof search

## What does it implement? (And how?)

- a formal logical system intended as foundation for mathematics
- decision procedures

# The Kepler Conjecture (year 1611)

*num pro apice, esto & alia cop*

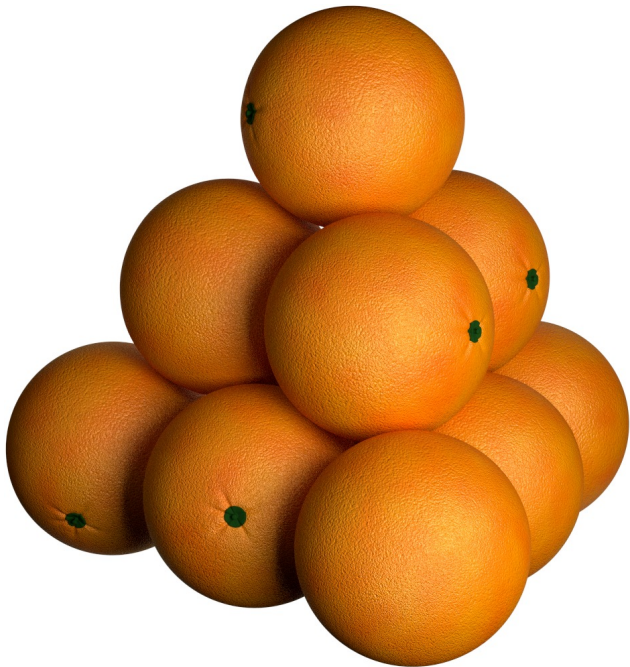


*alia  
ore  
Est  
fspa  
tam  
inti  
rior  
è sa  
qua  
bus  
pec  
sup  
tion  
gul  
Pat*

*necessitate concurrente cum ra*

The most compact way of stacking balls of the same size in space is a pyramid.

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



# The Kepler Conjecture (year 1611)

## Proved in 1998

- Tom Hales, 300 page proof using computer programs
- Submitted to the Annals of Mathematics

# The Kepler Conjecture (year 1611)

## Proved in 1998

- Tom Hales, 300 page proof using computer programs
- Submitted to the Annals of Mathematics
- 99% correct... but we cannot verify the programs

## 1039 equalities and inequalities

For example:

$$\frac{-x_1 x_3 - x_2 x_4 + x_1 x_5 + x_3 x_6 - x_5 x_6 + x_2(-x_2 + x_1 + x_3 - x_4 + x_5 + x_6)}{\sqrt{4x_2 \left( \begin{array}{l} x_2 x_4 (-x_2 + x_1 + x_3 - x_4 + x_5 + x_6) + \\ + x_1 x_5 (x_2 - x_1 + x_3 + x_4 - x_5 + x_6) + \\ + x_3 x_6 (x_2 + x_1 - x_3 + x_4 + x_5 - x_6) - \\ - x_1 x_3 x_4 - x_2 x_3 x_5 - x_2 x_1 x_6 - x_4 x_5 x_6 \end{array} \right)}} < \tan\left(\frac{\pi}{2} - 0.74\right)$$



# The Kepler Conjecture (year 1611)

## Solution? Formalized Proof!

- Formalize the proof using Proof Assistants
- Implement the computer code in the system
- Prove the code correct
- Run the programs inside the Proof Assistant

## Flyspeck Project

- Nearing Completion
- Many Proof Assistants and contributors

# Intel Pentium P5 (1994)

## FPU unit

- Division lookup table
- For certain inputs division result off

## Replacement

- Few customers cared, still 450M\$
- Birth of HOL Light
- Intel and AMD processors **formally verified**

```

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: "√(real 2) = real m / real n"
    and lowest_terms: "gcd m n = 1" ..
  from n_nonzero and sqrt_rat have "real m = √(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

```

```

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: "√(real 2) = real m / real n"
    and lowest_terms: "gcd m n = 1" ..
  from n_nonzero and sqrt_rat have "real m = √(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

```

## de Bruijn factor

## :: W The Irrationality of the Square Root of 2

theorem Th1:

for p being Element of NAT st p is prime holds  
sqrt p is irrational

proof

```
let p be Element of NAT ;
assume A1: p is prime ;
then A2: p > 1 by INT_2:def 4;
assume sqrt p is rational ;
then consider i being Integer, n being Element of NAT such that
A3: n <> 0 and
A4: sqrt p = i / n and
A5: for i1 being Integer
for n1 being Element of NAT st n1 <> 0 & sqrt p = i1 / n1 holds
n <= n1 by RAT_1:9;
A6: i = (sqrt p) * n by A3, A4, XCMPLX_1:87;
sqrt p >= 0 by SQUARE_1:def 2;
then reconsider m = i as Element of NAT by A6, INT_1:3;
A7: m ^2 = ((sqrt p) ^2) * (n ^2) by A6
.= p * (n ^2) by SQUARE_1:def 2 ;
then p divides m ^2 by NAT_D:def 3;
then p divides m by A1, NEWTON:80;
then consider m1 being Nat such that
A8: m = p * m1 by NAT_D:def 3;
n ^2 = (p * (p * (m1 ^2))) / p by A2, A7, A8, XCMPLX_1:89
.= p * (m1 ^2) by A2, XCMPLX_1:89 ;
then p divides n ^2 by NAT_D:def 3;
then p divides n by A1, NEWTON:80;
then consider n1 being Nat such that
A9: n = p * n1 by NAT_D:def 3;
A10: m1 / n1 = sqrt p by A2, A4, A8, A9, XCMPLX_1:91;
A11: n1 <> 0 by A3, A9;
then p * n1 > 1 * n1 by A2, XREAL_1:98;
hence contradiction by A5, A9, A11, A10;
end;
```

# Proof Assistant (2/2)

- Keep track of theories, definitions, assumptions
  - set up a theory that describes mathematical concepts (or models a computer system)
  - express logical properties of the objects
- Interaction - proof editing
  - typically interactive
  - specified theory and proofs can be edited
  - provides information about required proof obligations
  - allows further refinement of the proof
  - often manually providing a direction in which to proceed.
- Automation - proof search
  - various strategies
  - decision procedures
- Proof checking
  - checking of complete proofs
  - sometimes providing certificates of correctness
- Why should we trust it?
  - small core

# Can a Proof Assistant do all proofs?

## Decidability!

- Validity of formulas is undecidable
- (for non-trivial logical systems)

## Automated Theorem Provers

- Specific domains
- Adjust your problem
- Answers: Valid (Theorem with proof)
- Or: Countersatisfiable (Possibly with counter-model)

## Proof Assistants

- Generally applicable
- Direct modelling of problems
- Interactive

# Other Tools

## Computer Algebra

- Solving equations, simplifications, numerical approximations
- Maple, Mathematica, ...

## Model Checkers

- Space state abstraction
- Spin, Uppaal, ...

## ATPs

- Built in automation (model elimination, resolution)
- ACL2, Vampire, Eprover, SPASS, ...



# Users of Proof Assistants

## Computer Science

- Modelling and specifying systems
- Proving properties of systems
- Proving software correct

## Mathematics

- Defining concepts and theories
- Proving (mostly verifying) proofs
- (currently less common)

# Theorems and programs that use ITP

## Theorems

- Kepler Conjecture
- 4 color theorem
- Feit-Thomson theorem (finished last September)

## Software

- Processors and Chips
- Security Protocols
- Project Cristal (Comp-Cert)
- L4-Verified
- Java Bytecode

# History of Proof Assistants

## $\lambda$ -calculus (Church, 1940)

- Simple Type Theory
- Higher Order Logic

## Formulas as Types (Curry-Howard, de Bruijn)

- Proofs as Terms
- Reduce Proof Checking to Type Checking

## Automath

- First implementation

## LCF (Milner)

- ML programming language

# Multitude of Proof Assistants

## Characterized by various

- Foundations
- Interaction models
- Automation strategies
- Libraries
- Size of trusted core

## Examples

- HOL (HOL4, HOL-Light, ProofPower, HOL0), Mizar (and variants), PVS, Coq, Otter/Ivy, Isabelle/Isar (HOL, ZF, CTT, ...), Alfa/Agda, ACL2, IMPS, Metamath, Theorema, Lego, Nuprl,  $\Omega$ mega, B method, Minlog

# Coverage of Basic Mathematics

## Freek Wiedijk's list of 100 theorems

HOL Light	86
Mizar	57
Coq	49
Isabelle	49
ProofPower	42
PVS	16
nqthm/ACL2	16
NuPRL/MetaPRL	8
any	88

<http://www.cs.ru.nl/~freek/100/>

# Summary

## This Lecture

- What is a Proof Assistant
- Common Uses
- Comparison with other tools
- Formal proof examples
- De Bruijn factor
- History
- Characteristics
- Coverage of Basic Mathematics

## Next

- Typed  $\lambda$ -calculus
- HOL Light