# Interactive Theorem Provers

## from the perspective of Isabelle/Isar

Makarius Wenzel
Univ. Paris-Sud, LRI

July 2014

# 1 Introduction

# Notable ITP systems

**LISP based:**

**ACL2** `http://www.cs.utexas.edu/users/moore/acl2`

**PVS** `http://pvs.csl.sri.com`

**ML based:**

**HOL family:** HOL4, HOL-Light, ProofPower, . . .

**Coq** `http://coq.inria.fr`

**Isabelle/Isar** `http://isabelle.in.tum.de`

**Other:**

**Mizar** `http://www.mizar.org`

**Agda** `http://wiki.portal.chalmers.se/agda`

See also: *The Seventeen Provers of the World*, F. Wiedijk (ed.), LNAI 3600, 2006.

# The LCF family

**LCF**

- Edinburgh LCF (1979)
- Cambridge LCF (1985)

**HOL** (1984/1988)

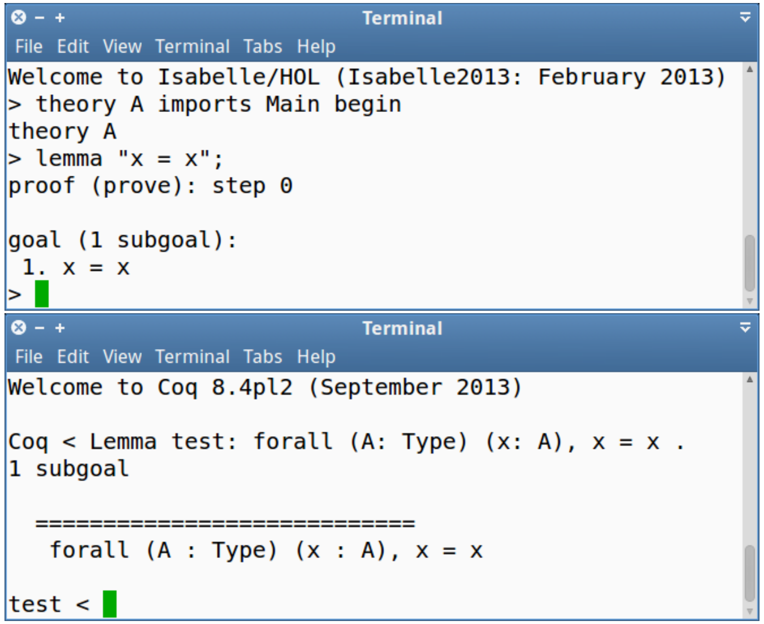
**Coq**

- Coc (1985/1988)
- ⋮
- Coq 8.4pl4 (May 2014)

**Isabelle**

- Isabelle (1986/1989)
- Isabelle/Isar (1999)
- ⋮
- Isabelle2013-2 (December 2013)

# TTY interaction

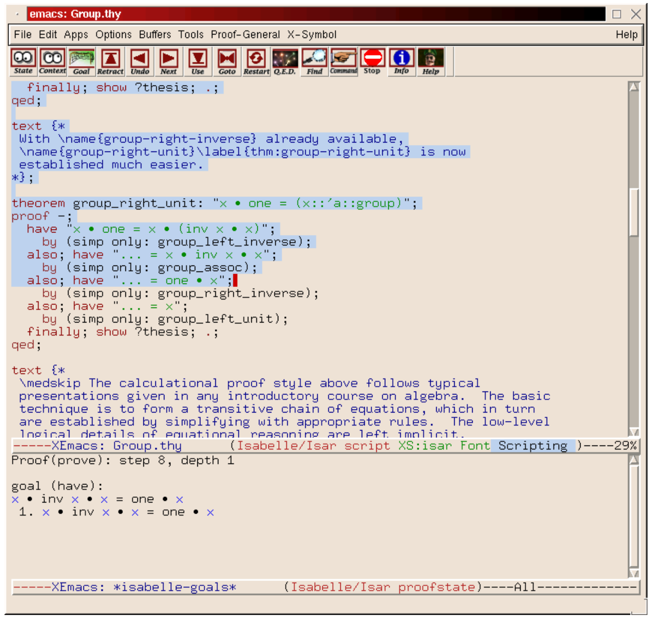


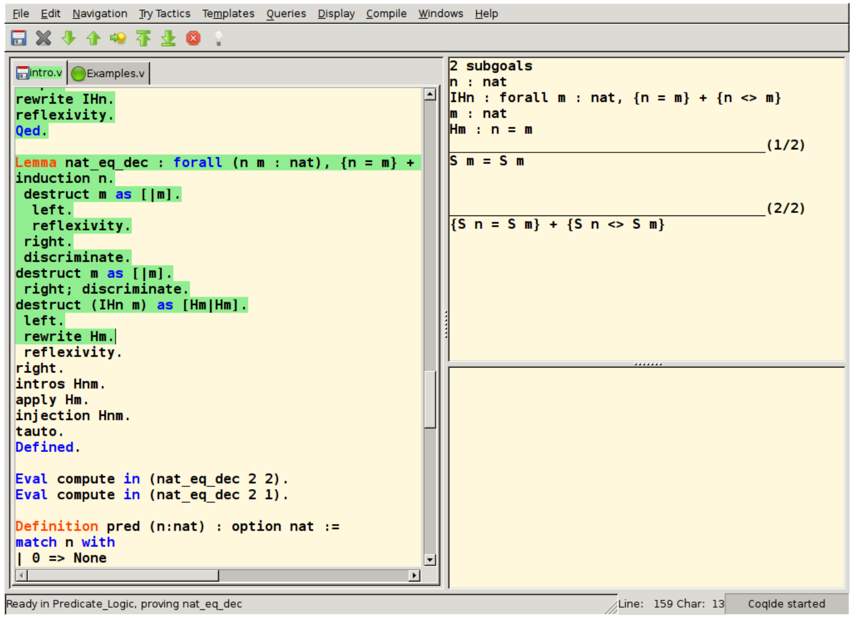

(Wikipedia: K. Thompson and D. Ritchie at PDP-11)

## Interaction model:

manual copy-paste from editor window into prover process

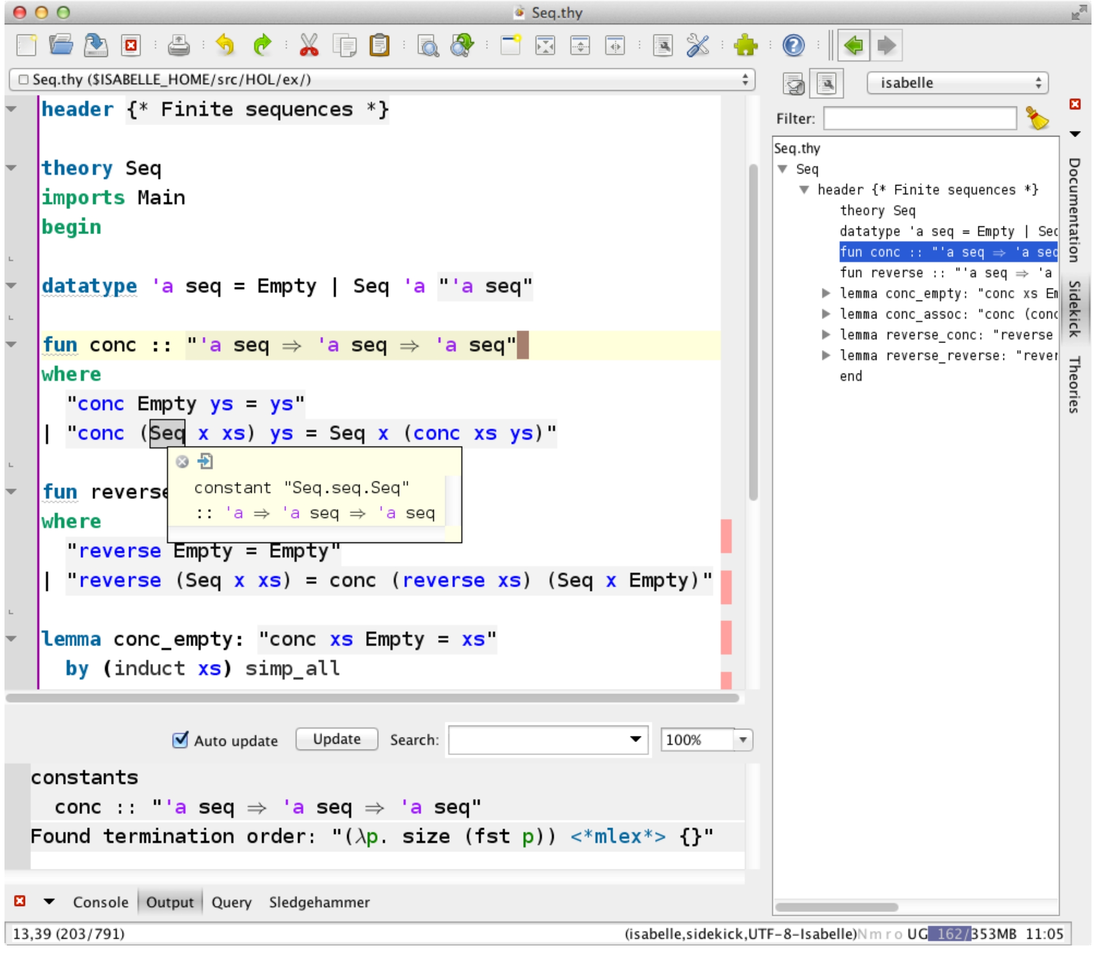
# Proof General (and clones)




**Interaction model:** automated copy-paste and undo in the editor, prover process in background

# Isabelle today: document-oriented interaction



# Example: functional specifications with proofs

**datatype** $'a\ seq = Empty \mid Seq\ 'a\ ('a\ seq)$

**fun** $concat :: 'a\ seq \Rightarrow 'a\ seq \Rightarrow 'a\ seq$
**where**
$\quad concat\ Empty\ ys = ys$
$\mid concat\ (Seq\ x\ xs)\ ys = Seq\ x\ (concat\ xs\ ys)$

**theorem** $concat\_empty$: $concat\ xs\ Empty = xs$
**by** $(induct\ xs)\ simp\_all$

**theorem** $conc\_assoc$: $concat\ (concat\ xs\ ys)\ zs = concat\ xs\ (concat\ ys\ zs)$
**by** $(induct\ xs)\ simp\_all$

# Example: unstructured proof "scripts"

**theorem** $concat\_empty'$: $concat\ xs\ Empty = xs$
  **apply** $(induct\ xs)$
   **apply** $simp$
  **apply** $simp$
  **done**

**theorem** $conc\_assoc'$: $concat\ (concat\ xs\ ys)\ zs = concat\ xs\ (concat\ ys\ zs)$
  **apply** $(induct\ xs)$
   **apply** $simp$
  **apply** $simp$
  **done**

## Example: abstract specifications and calculations

**class** *group* $=$ *times* $+$ *one* $+$ *inverse* $+$
  **assumes** *group_assoc*: $(x * y) * z = x * (y * z)$
    **and** *group_left_one*: $1 * x = x$
    **and** *group_left_inverse*: *inverse* $x * x = 1$

**theorem** (**in** *group*) *group_right_inverse*: $x *$ *inverse* $x = 1$
  $\langle$*proof*$\rangle$

**theorem** (**in** *group*) *group_right_one*: $x * 1 = x$
**proof** $-$
  **have** $x * 1 = x * ($*inverse* $x * x)$ **by** (*simp only*: *group_left_inverse*)
  **also have** $\ldots = x *$ *inverse* $x * x$ **by** (*simp only*: *group_assoc*)
  **also have** $\ldots = 1 * x$ **by** (*simp only*: *group_right_inverse*)
  **also have** $\ldots = x$ **by** (*simp only*: *group_left_one*)
  **finally show** *?thesis* **.**
**qed**

# 2 Proof Systems

# Isabelle/Pure: formal context

**Logical judgement:**

$$\boxed{\Theta, \Gamma \vdash \varphi}$$

- background theory $\Theta$
  (polymorphic types, constants, axioms; global data)
- proof context $\Gamma$ (fixed variables, assumptions; local data)

**Operations on theories:**

- merge and extend: $\Theta_3 = \Theta_1 \cup \Theta_2 + \tau + c :: \tau + c \equiv t$
- symbolic sub-theory relation: $\Theta_1 \subseteq \Theta_2$
- transfer of results: if $\Theta_1 \subseteq \Theta_2$ and $\Theta_1, \Gamma \vdash \varphi$ then $\Theta_2, \Gamma \vdash \varphi$

# Isabelle/Pure: primitive inferences

**Syntax (types and terms):**

| | |
|---|---|
| $fun :: (type, type)type$ | function space $'a \Rightarrow 'b$ |
| $all :: ('a \Rightarrow prop) \Rightarrow prop$ | universal quantification $\bigwedge x.\ B\ x$ |
| $imp :: prop \Rightarrow prop \Rightarrow prop$ | implication $A \Longrightarrow B$ |

**Derivations (theorems):** implicit theory $\Theta$

$$\frac{A \in \Theta}{\vdash A}\,(axiom) \qquad \frac{}{A \vdash A}\,(assume)$$

$$\frac{\Gamma \vdash B[x] \quad x \notin \Gamma}{\Gamma \vdash \bigwedge x.\ B[x]}\,(\bigwedge\text{-}intro) \qquad \frac{\Gamma \vdash \bigwedge x.\ B[x]}{\Gamma \vdash B[a]}\,(\bigwedge\text{-}elim)$$

$$\frac{\Gamma \vdash B}{\Gamma - A \vdash A \Longrightarrow B}\,(\Longrightarrow\text{-}intro) \qquad \frac{\Gamma_1 \vdash A \Longrightarrow B \quad \Gamma_2 \vdash A}{\Gamma_1 \cup \Gamma_2 \vdash B}\,(\Longrightarrow\text{-}elim)$$

# Isabelle/Isar: block-structured reasoning

## Universal context: fix and assume

{
  **fix** $x$
  **have** $B\ x\ \langle proof \rangle$
}
**have** $\bigwedge x.\ B\ x$ **by** $fact$

{
  **assume** $A$
  **have** $B\ \langle proof \rangle$
}
**have** $A \Longrightarrow B$ **by** $fact$

## Existential context: obtain

{
  **obtain** $a$ **where** $B\ a\ \langle proof \rangle$
  **have** $C\ \langle proof \rangle$
}
**have** $C$ **by** $fact$

# 3 Proof Search

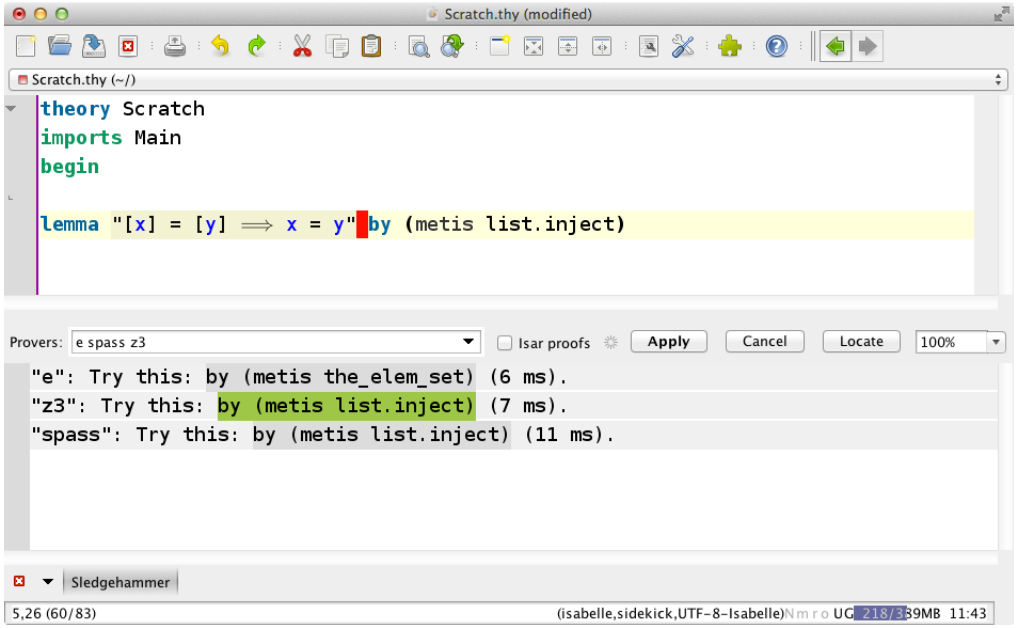
# Isabelle/HOL proof methods

- $rule$: generic Natural Deduction (with HO unification)
- $cases$: elimination, syntactic representation of datatypes, inversion of inductive sets and predicates
- $induct$ and $coinduct$: induction and coinduction of types, sets, predicates
- $simp$: equational reasoning by the Simplifier (HO rewriting), with possibilities for add-on tools
- $fast$ and $blast$: classical reasoning (tableau)
- $auto$ and $force$: combined simplification and classical reasoning
- $arith$, $presburger$: specific theories
- $smt$: Z3 with proof reconstruction

# Sledgehammer

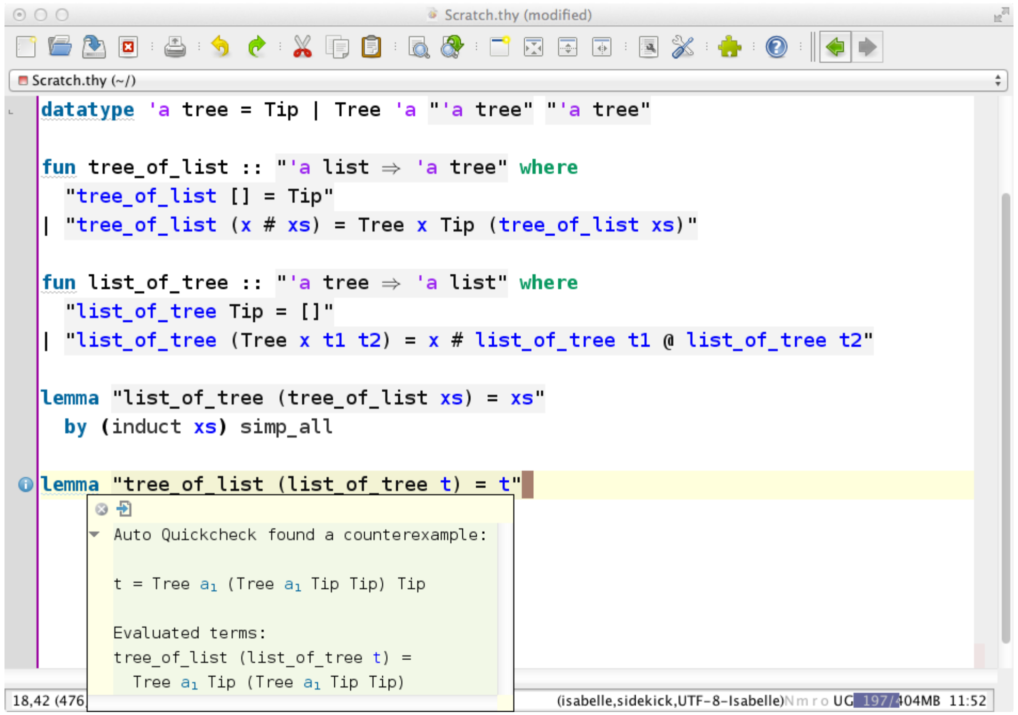
## Idea:

- heavy external ATPs / SMTs for proof search
- light internal ATP (Metis) for proof reconstruction



# Automated disprovers — counter examples

- **quickcheck** based on random functional evaluation
- **nitpick** based on relational model finder



# 4 Proof Formats

# Proof formats: open-ended, no standards

### De-facto formats:

**LCF and HOL:** ML source as input and output

**Coq:** tactic scripts, e.g. Ltac, SSReflect

**Isabelle/Isar:**

- structured proof documents (Isar language)
- unstructured apply scripts (tactic emulation)

### General LCF approach:

use ML to implement your own application-specific proof formats

# 5 Proof Production

# The “LCF approach”

**Correctness by construction:** (R. Milner, 1979)

1. abstract datatype $thm$ in ML (the “meta language”), constructors are the rules of the logic (the “object language”)
2. implementation of arbitrary proof tools in ML, with explicit $thm$ construction at run-time

**Notes:**

- need to distinguish proof search from actual $thm$ inferences
- $thm$ values are abstract: proofs are not stored in memory, but: optional proof trace or proof term
- goal-directed LCF-approach fits well to shared-memory multiprocessing (multicore hardware)

# 6 Proof Consumption

# Proof consumption in Isabelle/HOL

**HOL-Light importer:**
replay of primitive inferences from other LCF-kernel (huge trace)

**SMT proof method:**
connection to Z3, with proof reconstruction by standard proof tools of Isabelle/HOL: $simp$, $blast$, $auto$ etc.

**Sledgehammer:**

- heavy external ATPs / SMTs for proof search
- light internal ATP (Metis) for proof reconstruction

# 7 Proof Applications

# Big formalization projects

**Flyspeck** `https://code.google.com/p/flyspeck` (T. Hales, HOL-Light): formal proof of Kepler's Conjecture

**L4.verified** `http://ertos.nicta.com.au/research/l4.verified` (G. Klein, Isabelle/HOL): formally correct operating system kernel

**Feit-Thompson Odd Order Theorem** `http://www.msr-inria.fr/news/feit-thomson-proved-in-coq` (G. Gonthier, Coq/SSReflect)

**CompCert verified compiler** `http://compcert.inria.fr/doc` (X. Leroy, Coq): optimizing C-compiler for various assembly languages, written and proven in the functional language of Coq

# Libraries of formalized mathematics

## Archive of Formal Proofs (AFP)

`http://afp.sf.net`

Isabelle/HOL

## Mathematical Components

`http://www.msr-inria.fr/projects/mathematical-components-2`

Coq/SSReflect

## Mizar Mathematical Library

`http://www.mizar.org/library`

Mizar

# 8 Conclusions

# What is ITP? What is Isabelle/Isar?


Hanabusa Itchō: “Blind monks examining an elephant”

## Helpful hints

**New users:**

- Spend time to develop a sense for more than one accidental candidate, before making a commitment.
- Spend substantial time to become proficient with the system of your choice.

**Old users:**

- Learn how other proof assistants work, and what are their specific strengths and weaknesses.

**Isabelle users:**

- Submit your finished applications to AFP `http://afp.sf.net`

Happy proving!