# Higher-Order Logic (HOL) as a Lingua Franca for Argumentative Reasoning Agents

**David Fuenmayor** & Christoph Benzmüller
AI Systems Engineering (U. Bamberg)

10th Conference on Artificial Intelligence and Theorem Proving (AITP 2025)
August 31 - September 5, 2025, Aussois, France

# The “Universal Logical Reasoning” Programme

*“Classical higher-order logic, when utilized as a meta-logic in which various other (classical and non-classical) logics can be shallowly embedded, is well suited for realising a universal logic reasoning approach. Universal logic reasoning in turn, as envisioned already by Leibniz, may support the rigorous formalisation and deep logical analysis of rational arguments within machines.“*

**Calculemus!**

Benzmüller (2017) “Universal Reasoning, Rational Argumentation and Human-Machine Interaction”

# Main Idea: HOL as universal meta-logic

cf. Benzmüller (2019) Universal (Meta-)Logical Reasoning: Recent Successes

2 BASIC MODAL LOGIC

In this section we introduce the basic modal language and its relational semantics. We define basic modal syntax, introduce models and frames, and give the satisfaction definition. We then draw the reader's attention to the internal perspective that modal languages offer on relational structure, and explain why models and frames should be thought of as graphs. Following this we give the standard translation. This enables us to convert any basic modal formula into a first-order formula with one free variable. The standard translation is a bridge between the modal and classical worlds, a bridge that underlies much of the work of this chapter.

*2.1 First steps in relational semantics*

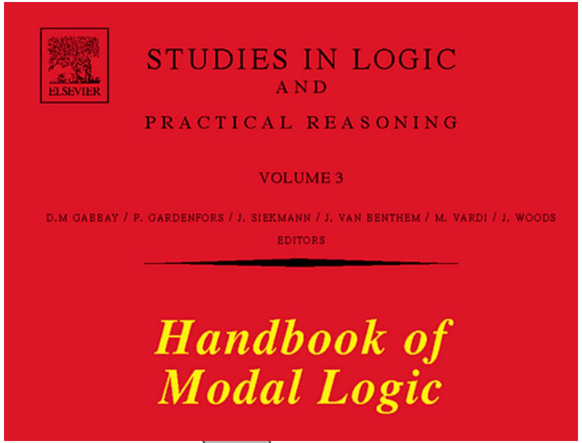
**Syntax**

**Metalanguage**

…ose elements we typically write as $p, q, r$ and …ents we typically write as $m, m', m''$, and so …*nature* (*or similarity type*) of the language; in …what follows we'll tacitly assume that PROP is denumerably infinite, and we'll often work with signatures in which MOD contains only a single element. Given a signature, we define the *basic modal language* (over the signature) as follows:

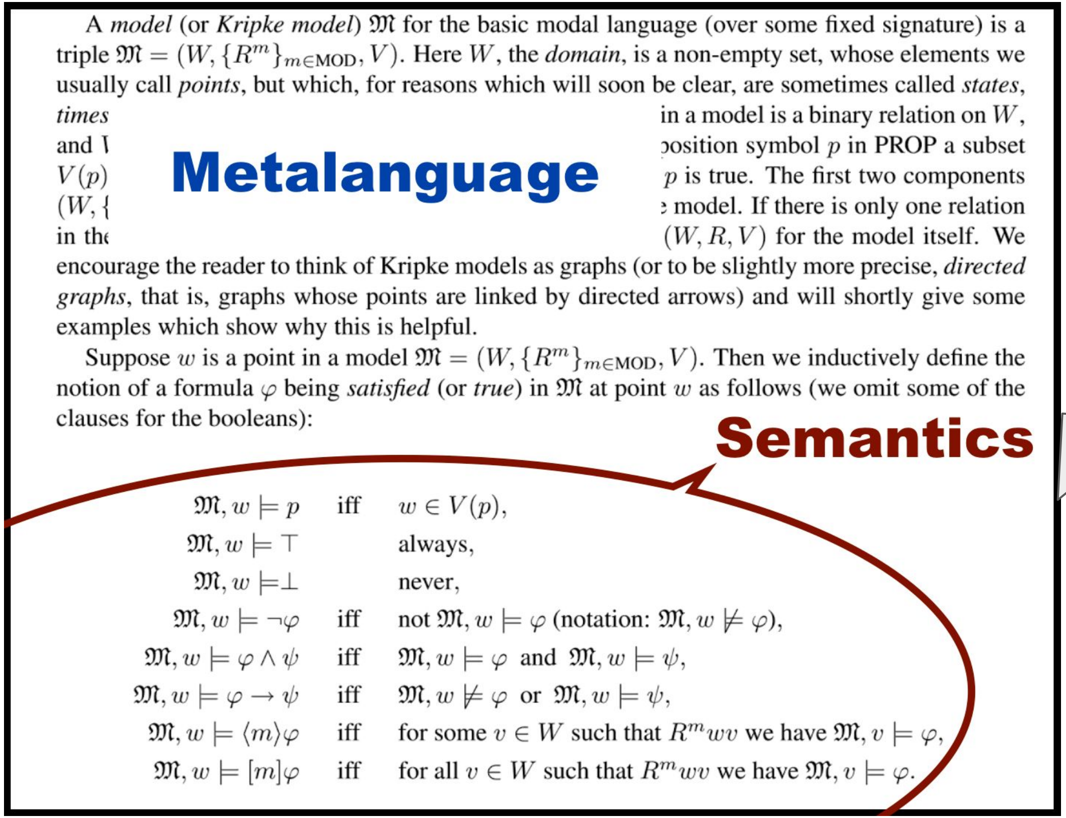
$$\varphi \quad ::= \quad p \mid \top \mid \bot \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \to \psi \mid \varphi \leftrightarrow \psi \mid \langle m \rangle \varphi \mid [m]\varphi.$$

That is, a basic modal formula is either a proposition symbol, a boolean constant, a boolean combination of basic modal formulas, or (most interesting of all) a formula prefixed by a diamond



# Main Idea: HOL as universal meta-logic

cf. Benzmüller (2019) Universal (Meta-)Logical Reasoning: Recent Successes

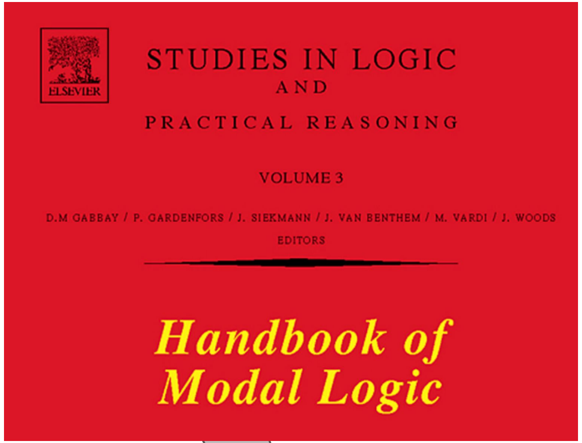


A *model* (or *Kripke model*) $\mathfrak{M}$ for the basic modal language (over some fixed signature) is a triple $\mathfrak{M} = (W, \{R^m\}_{m \in \mathrm{MOD}}, V)$. Here $W$, the *domain*, is a non-empty set, whose elements we usually call *points*, but which, for reasons which will soon be clear, are sometimes called *states*, *times* … in a model is a binary relation on $W$, and … position symbol $p$ in PROP a subset $V(p)$ … $p$ is true. The first two components $(W, \{$ … model. If there is only one relation in the … $(W, R, V)$ for the model itself. We encourage the reader to think of Kripke models as graphs (or to be slightly more precise, *directed graphs*, that is, graphs whose points are linked by directed arrows) and will shortly give some examples which show why this is helpful.

Suppose $w$ is a point in a model $\mathfrak{M} = (W, \{R^m\}_{m \in \mathrm{MOD}}, V)$. Then we inductively define the notion of a formula $\varphi$ being *satisfied* (or *true*) in $\mathfrak{M}$ at point $w$ as follows (we omit some of the clauses for the booleans):

| | | |
|---|---|---|
| $\mathfrak{M}, w \models p$ | iff | $w \in V(p)$, |
| $\mathfrak{M}, w \models \top$ | | always, |
| $\mathfrak{M}, w \models \bot$ | | never, |
| $\mathfrak{M}, w \models \neg\varphi$ | iff | not $\mathfrak{M}, w \models \varphi$ (notation: $\mathfrak{M}, w \not\models \varphi$), |
| $\mathfrak{M}, w \models \varphi \wedge \psi$ | iff | $\mathfrak{M}, w \models \varphi$ and $\mathfrak{M}, w \models \psi$, |
| $\mathfrak{M}, w \models \varphi \to \psi$ | iff | $\mathfrak{M}, w \not\models \varphi$ or $\mathfrak{M}, w \models \psi$, |
| $\mathfrak{M}, w \models \langle m \rangle\varphi$ | iff | for some $v \in W$ such that $R^m wv$ we have $\mathfrak{M}, v \models \varphi$, |
| $\mathfrak{M}, w \models [m]\varphi$ | iff | for all $v \in W$ such that $R^m wv$ we have $\mathfrak{M}, v \models \varphi$. |





# Shallow (Semantical) Embedding in HOL

HOL (meta-logic) $\varphi$ ::=

L (object-logic) $\psi$ ::=

Embedding of  in 

=

=

=

=

=

6

# Shallow (Semantical) Embedding in HOL

HOL $s, t ::= c_\alpha \mid x_\alpha \mid (\lambda x_\alpha s_\beta)_{\alpha\to\beta} \mid (s_{\alpha\to\beta}\, t_\alpha)_\beta \mid \neg s_o \mid s_o \vee t_o \mid \forall x_\alpha\, t_o$

HOML $\varphi, \psi ::= \ldots \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \to \psi \mid \Box\varphi \mid \Diamond\varphi \mid \forall x_\gamma\, \varphi \mid \exists x_\gamma\, \varphi$

HOML in HOL: HOML formulas $\varphi$ are mapped to HOL predicates $\varphi_{\mu\to o}$ (explicit representation of labelled formulas)

$$\neg = \lambda\varphi_{\mu\to o}\lambda w_\mu \neg\varphi w$$
$$\wedge = \lambda\varphi_{\mu\to o}\lambda\psi_{\mu\to o}\lambda w_\mu(\varphi w \wedge \psi w)$$
$$\to = \lambda\varphi_{\mu\to o}\lambda\psi_{\mu\to o}\lambda w_\mu(\neg\varphi w \vee \psi w)$$
$$\forall = \lambda h_{\gamma\to(\mu\to o)}\lambda w_\mu \forall d_\gamma\, hdw$$
$$\exists = \lambda h_{\gamma\to(\mu\to o)}\lambda w_\mu \exists d_\gamma\, hdw$$
$$\Box = \lambda\varphi_{\mu\to o}\lambda w_\mu \forall u_\mu\, (\neg rwu \vee \varphi u)$$
$$\Diamond = \lambda\varphi_{\mu\to o}\lambda w_\mu \exists u_\mu\, (rwu \wedge \varphi u)$$
$$\text{valid} = \lambda\varphi_{\mu\to o}\forall w_\mu \varphi w$$

Ax (polymorphic over $\gamma$)

The equations in Ax are given as axioms to the HOL provers!

# Shallow (Semantical) Embedding in HOL

HOL $s, t \quad ::= \quad c_\alpha \mid x_\alpha \mid (\lambda x_\alpha s_\beta)_{\alpha \to \beta} \mid (s_{\alpha \to \beta}\, t_\alpha)_\beta \mid \neg s_o \mid s_o \vee t_o \mid \forall x_\alpha\, t_o$

HOML $\varphi, \psi \quad ::= \quad \ldots \mid \neg \varphi \mid \varphi \wedge \psi \mid \varphi \to \psi \mid \Box \varphi \mid \Diamond \varphi \mid \forall x_\gamma\, \varphi \mid \exists x_\gamma\, \varphi$

HOML in HOL: HOML formulas $\varphi$ are mapped to HOL predicates $\varphi_{\mu \to o}$ (explicit representation of labelled formulas)

| | | |
|---|---|---|
| $\neg$ | = | $\lambda \varphi_{\mu \to o} \lambda w_\mu \neg \varphi w$ |
| $\wedge$ | = | $\lambda \varphi_{\mu \to o} \lambda \psi_{\mu \to o} \lambda w_\mu (\varphi w \wedge \psi w)$ |
| $\to$ | = | $\lambda \varphi_{\mu \to o} \lambda \psi_{\mu \to o} \lambda w_\mu (\neg \varphi w \vee \psi w)$ |
| $\forall$ | = | $\lambda h_{\gamma \to (\mu \to o)} \lambda w_\mu \forall d_\gamma\, h d w$ |
| $\exists$ | = | $\lambda h_{\gamma \to (\mu \to o)} \lambda w_\mu \exists d_\gamma\, h d w$ |
| $\Box$ | = | $\lambda \varphi_{\mu \to o} \lambda w_\mu \forall u_\mu\, (\neg \mathbf{r} w u \vee \varphi u)$ |
| $\Diamond$ | = | $\lambda \varphi_{\mu \to o} \lambda w_\mu \exists u_\mu\, (\mathbf{r} w u \wedge \varphi u)$ |
| valid | = | $\lambda \varphi_{\mu \to o} \forall w_\mu \varphi w$ |

C. Benzmüller & L. Paulson (2013) "Quantified Multimodal Logics in Simple Type Theory" Logica Universalis

The equations in Ax are given as axioms to the HOL provers!

# Shallow (Semantical) Embedding in Isabelle/HOL

```
consts aRel::"w⇒w⇒bool" (infixr "r")
abbreviation mbox :: "(w⇒bool)⇒(w⇒bool)" ("□_") where "□φ ≡ (λw. ∀v. w r v ⟶ φ v)"
abbreviation mdia :: "(w⇒bool)⇒(w⇒bool)" ("◇_") where "◇φ ≡ (λw. ∃v. w r v  ∧  φ v)"
```

```
abbreviation mnot::"(w⇒bool)⇒(w⇒bool)" ("¬_") where "¬φ ≡ λw. ¬(φ w)"
abbreviation mand::"(w⇒bool)⇒(w⇒bool)⇒(w⇒bool)" (infix "∧") where "φ∧ψ ≡ λw. (φ w)∧(ψ w)"
abbreviation mor::"(w⇒bool)⇒(w⇒bool)⇒(w⇒bool)" (infix "∨") where "φ∨ψ ≡ λw. (φ w)∨(ψ w)"
abbreviation mimp::"(w⇒bool)⇒(w⇒bool)⇒(w⇒bool)" (infix "→") where "φ→ψ ≡ λw.(φ w)⟶(ψ w)"
```

```
consts Actualized::"e⇒w⇒bool" (infix "actualizedAt")

abbreviation mforallAct::"(e⇒w⇒bool)⇒(w⇒bool)" ("∀ᴬ")
  where "∀ᴬΦ ≡ λw. ∀x. (x actualizedAt w)⟶(Φ x w)"
abbreviation mexistsAct::"(e⇒w⇒bool)⇒(w⇒bool)" ("∃ᴬ")
  where "∃ᴬΦ ≡ λw. ∃x. (x actualizedAt w) ∧ (Φ x w)"
```

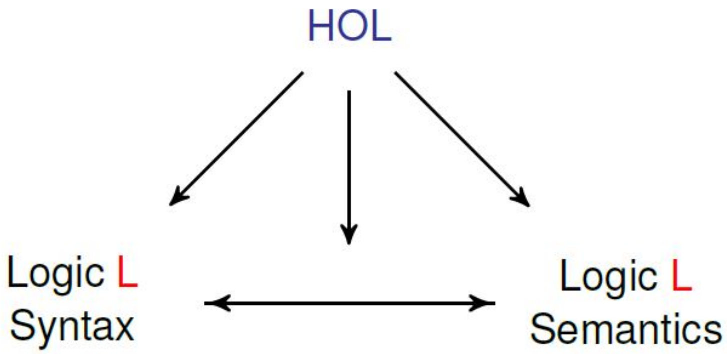
```
axiomatization where
T: "⌊∀φ. □φ → φ⌋" and (* or simply T: "reflexive aRel"*)
B: "⌊∀φ. φ →  □◇φ⌋" and (* or simply T: "symmetric aRel"*)
IV: "⌊∀φ. □φ →  □□φ⌋" and (* or simply T: "transitive aRel"*)
V: "⌊∀φ. ◇φ → □◇φ⌋" (* or simply T: "euclidean aRel"*)
```

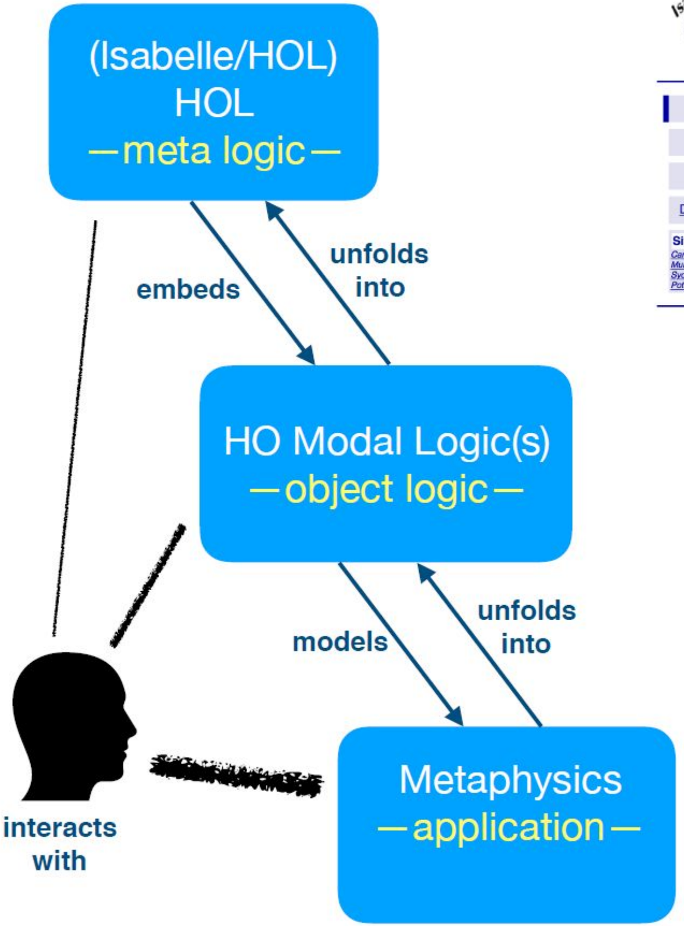
We can combine logics by adding/removing (meta-)axioms and definitions in the embedding logic.

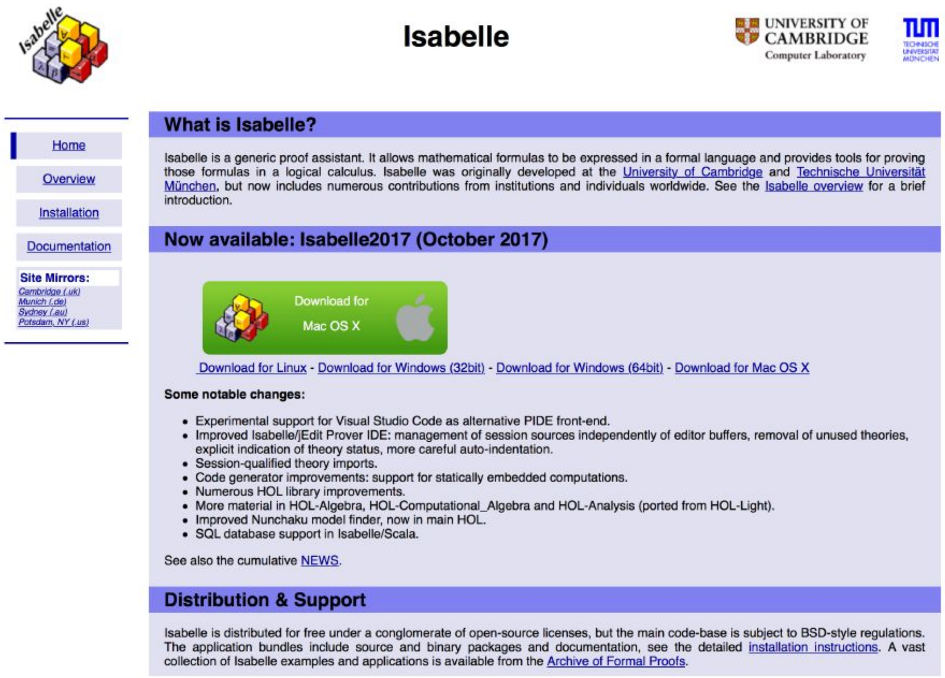
# Embedding Non-Classical Logics in HOL

Logics **L** embedded in HOL (with quantifiers!)

- Multi-modal & hybrid logics
- **Deontic logics** & conditional logics
- Many-valued logics
- Free logics (e.g. for category theory)
- 2D-semantics (Kaplan's Logic of Indexicals)
- Dynamic logics (incl. logics of preference & **public announcement logics**)
- **paraconsistent logics** & paracomplete logics
- **Substructural logics** (Lambek calculus, relevance logics, linear logics, etc.)





Isabelle

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Home
Overview
Installation
Documentation

Site Mirrors:
Cambridge (.uk)
Munich (.de)
Sydney (.au)
Potsdam, NY (.us)

## What is Isabelle?

Isabelle is a generic proof assistant. It allows mathematical formulas to be expressed in a formal language and provides tools for proving those formulas in a logical calculus. Isabelle was originally developed at the University of Cambridge and Technische Universität München, but now includes numerous contributions from institutions and individuals worldwide. See the Isabelle overview for a brief introduction.

## Now available: Isabelle2017 (October 2017)

Download for Mac OS X

Download for Linux - Download for Windows (32bit) - Download for Windows (64bit) - Download for Mac OS X

**Some notable changes:**

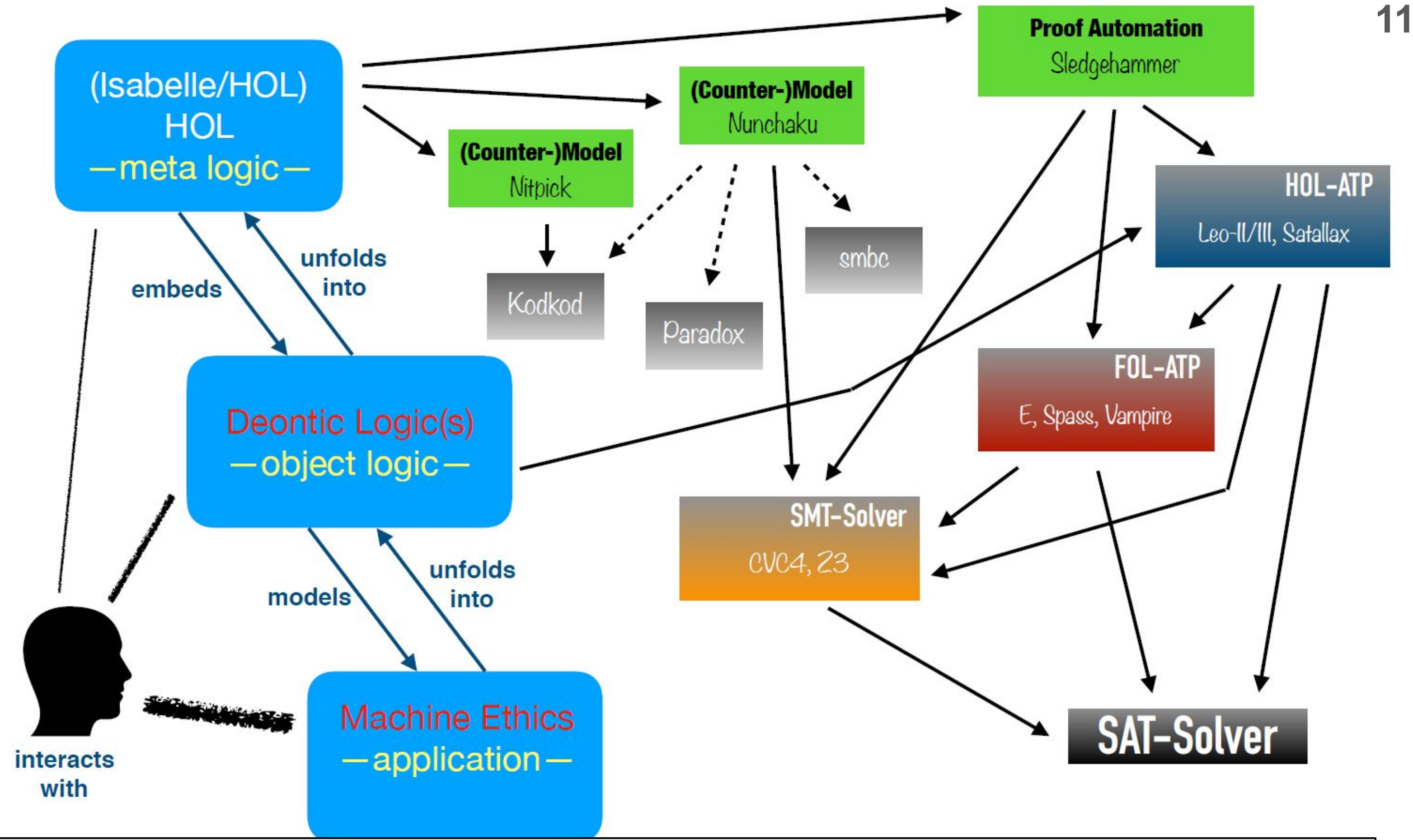
- Experimental support for Visual Studio Code as alternative PIDE front-end.
- Improved Isabelle/jEdit Prover IDE: management of session sources independently of editor buffers, removal of unused theories, explicit indication of theory status, more careful auto-indentation.
- Session-qualified theory imports.
- Code generator improvements: support for statically embedded computations.
- Numerous HOL library improvements.
- More material in HOL-Algebra, HOL-Computational_Algebra and HOL-Analysis (ported from HOL-Light).
- Improved Nunchaku model finder, now in main HOL.
- SQL database support in Isabelle/Scala.

See also the cumulative NEWS.

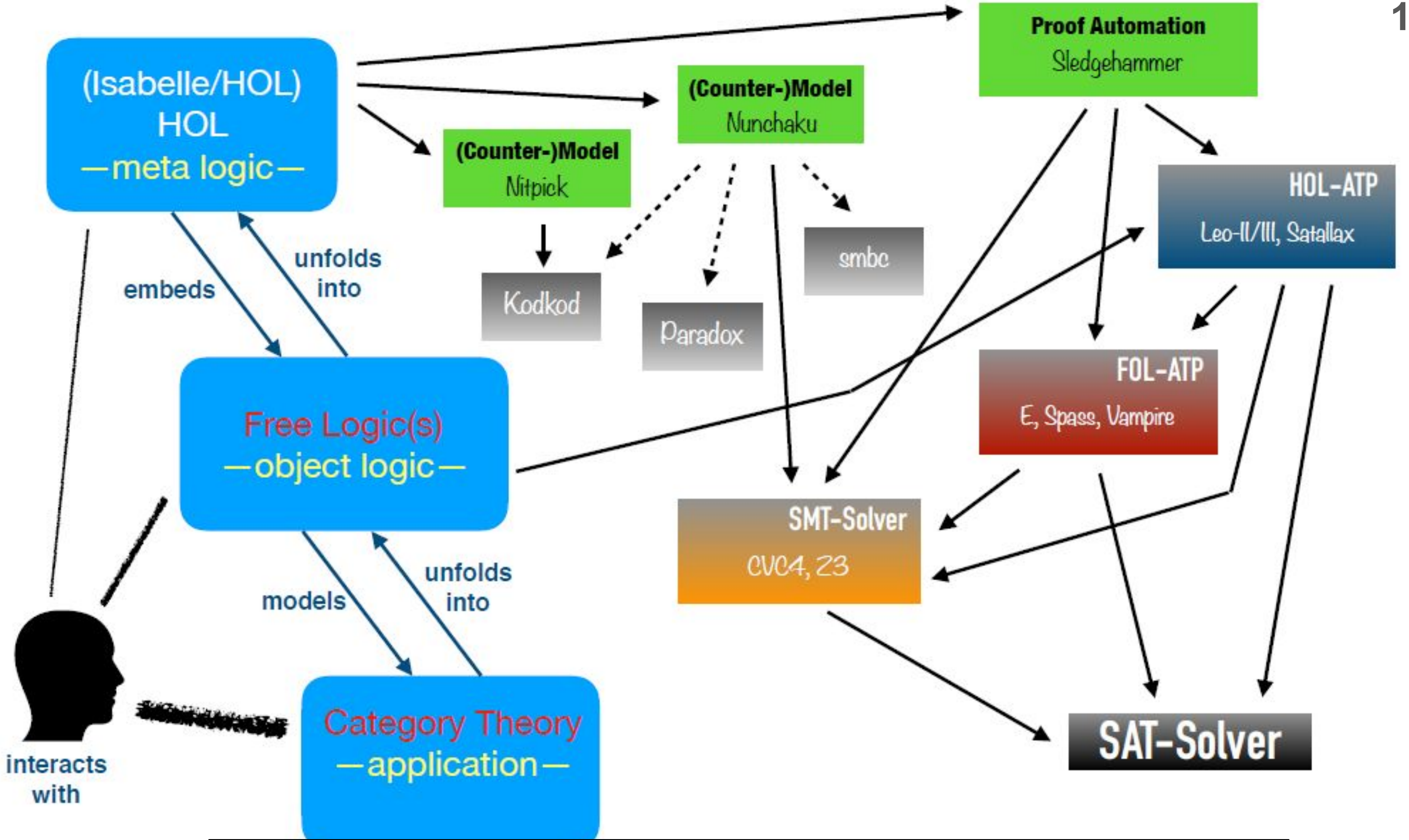
## Distribution & Support

Isabelle is distributed for free under a conglomerate of open-source licenses, but the main code-base is subject to BSD-style regulations. The application bundles include source and binary packages and documentation, see the detailed installation instructions. A vast collection of Isabelle examples and applications is available from the Archive of Formal Proofs.

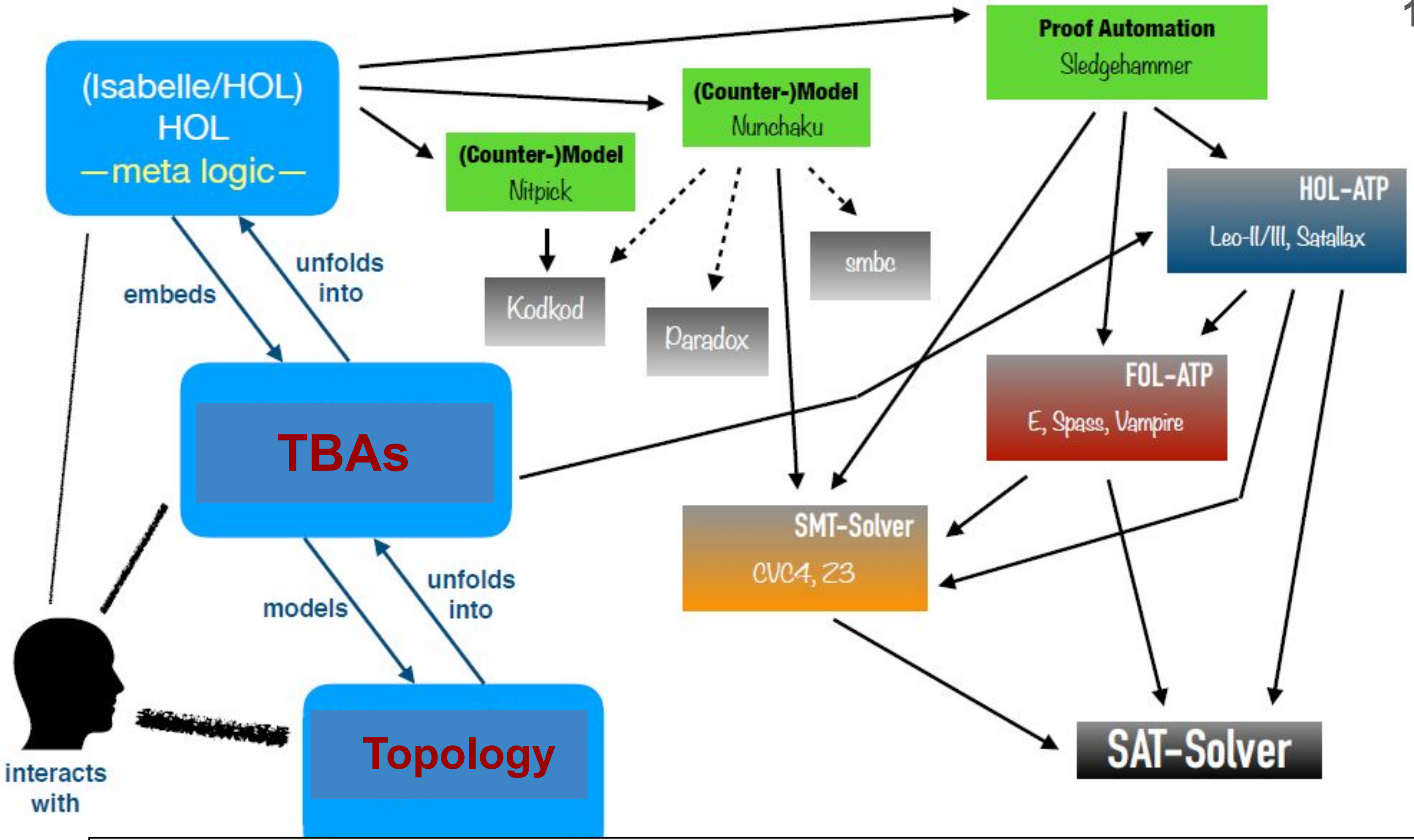
D. Kirschner, C. Benzmüller & E. Zalta "Computer Science and Metaphysics: A Cross-Fertilization" Open Philosophy, 2 (2019)




D. Fuenmayor, C. Benzmüller (2020) "Normative Reasoning with Expressive Logic Combinations"

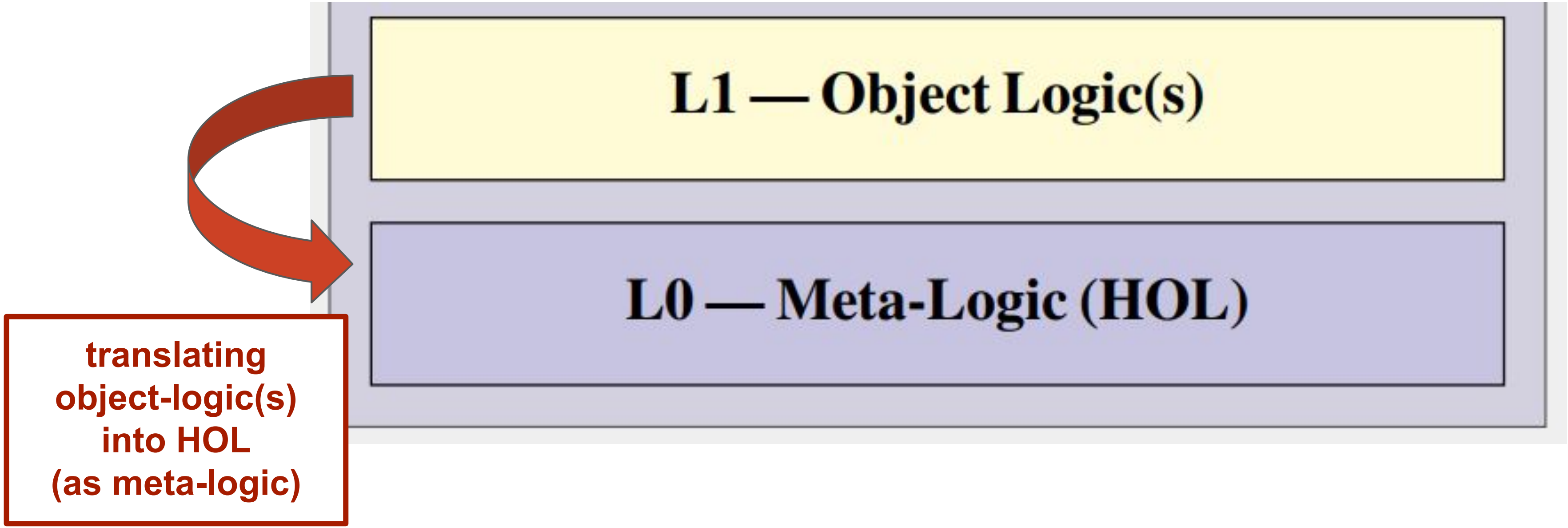



Benzmüller (2019) “Universal (meta-)logical reasoning: Recent successes”

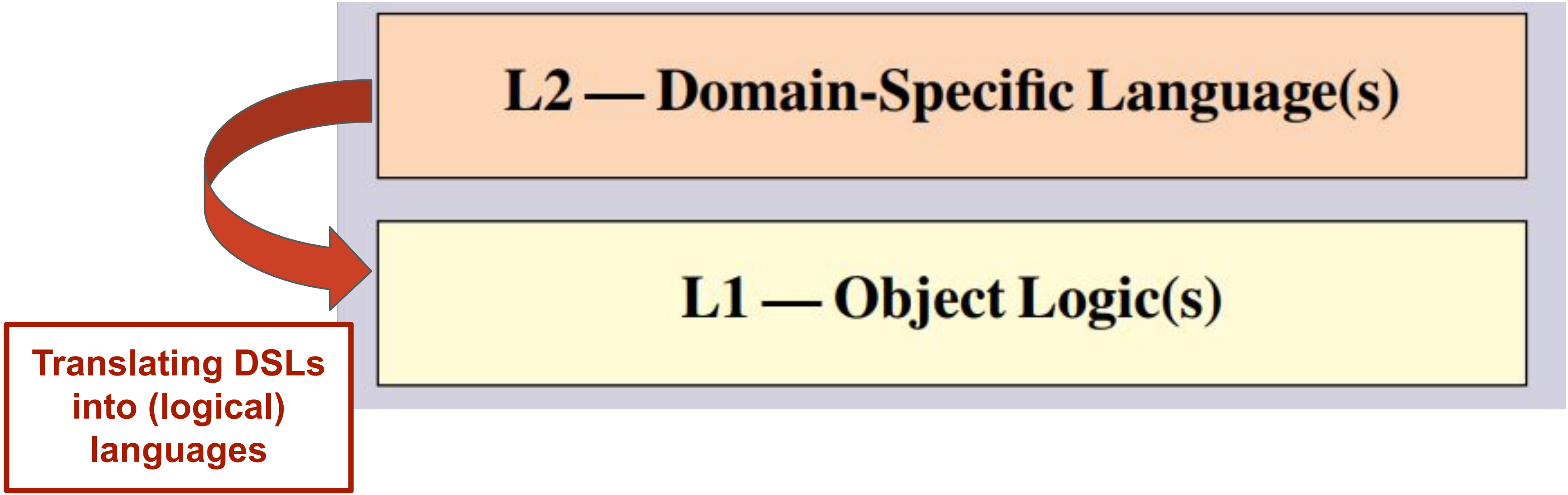



D. Fuenmayor, F. Serrano (2022) “Formalising Basic Topology for Computational Logic in Simple Type Theory” https://link.springer.com/chapter/10.1007/978-3-031-16681-5_4

# HOL as Universal Meta-Logic



# HOL as Universal Meta-Logic



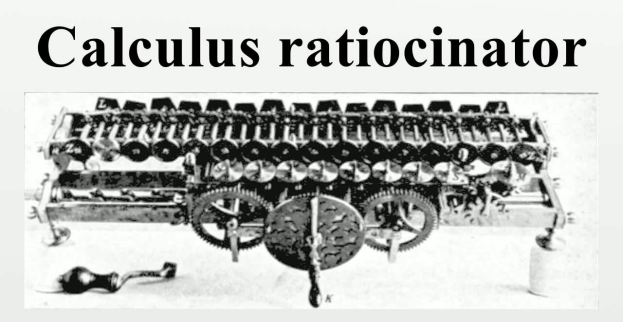
# The Universal (?) Logical Reasoning Programme

### *Leibniz's "Calculemus"*

*"... if controversies were to arise, there would be no more need of disputation between two philosophers than between two accountants. For it would suffice for them to take their pencils in their hands and to sit down at the abacus, and say to each other: Calculemus."*

**characteristica universalis**

$$(\forall x.(P(x) \land Q(x)) \leftrightarrow ((\forall x.P(x)) \land (\forall x.Q(x)))$$
$$(\exists x.(P(x) \land Q(x)) \to ((\exists x.P(x)) \land (\exists x.Q(x)))$$
$$(\exists x.(P(x) \lor Q(x)) \leftrightarrow ((\exists x.P(x)) \lor (\exists x.Q(x)))$$
$$((\forall x.P(x)) \lor (\forall x.Q(x))) \to (\forall x.(P(x) \lor Q(x))$$
$$(\exists x.\forall y.R(x,y)) \to (\forall y.\exists x.R(x,y))$$
$$(\neg(\exists x.P(x))) \leftrightarrow (\forall x.(\neg P(x))$$
$$(\neg(\forall x.P(x))) \leftrightarrow (\exists x.(\neg P(x))$$
$$(\neg(\exists xpt.P(x))) \leftrightarrow (\forall xpt.(\neg P(x))$$
$$(\neg(\forall xpt.P(x))) \leftrightarrow (\exists xpt.(\neg P(x))$$
$$(\forall x.(x = t \to F(x))) \leftrightarrow F(t)$$
$$(\exists x.(x = t \land F(x))) \leftrightarrow F(t)$$



# The Universal and Pluralistic Logical Reasoning Programme

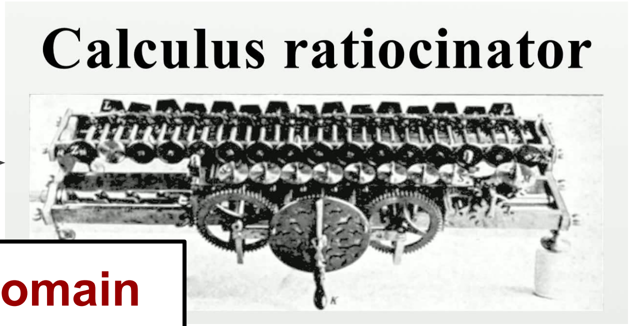
### *Leibniz's "Calculemus"*

*"... if controversies were to arise, there would be no more need of disputation between two philosophers than between two accountants. For it would suffice for them to take their pencils in their hands and to sit down at the abacus, and say to each other: Calculemus."*

**characteristica universalis**

$$(\forall x.(P(x) \wedge Q(x)) \leftrightarrow ((\forall x.P(x)) \wedge (\forall x.Q(x)))$$
$$(\exists x.(P(x) \wedge Q(x)) \rightarrow ((\exists x.P(x)) \wedge (\exists x.Q(x)))$$
$$(\exists x.(P(x) \vee Q(x)) \leftrightarrow ((\exists x.P(x)) \vee (\exists x.Q(x)))$$
$$((\forall x.P(x)) \vee (\forall x.Q(x))) \rightarrow (\forall x.(P(x) \vee Q(x))$$
$$(\exists x.\forall y.R(x,y)) \rightarrow (\forall y.\exists x.R(x,y))$$
$$(\neg(\exists x.P(x))) \leftrightarrow (\forall x.(\neg P(x))$$

**Domain specific!**




**Domain specific!**

**+ EXPLANATION**

**Domain specific!**

# A Digression:

Artificial Intelligence
Volume 287, October 2020, 103348

ELSEVIER

## Designing normative theories for ethical and legal reasoning: LogiKEy framework, methodology, and tool support ☆

Christoph Benzmüller [b, a], Xavier Parent [a], Leendert van der Torre [a, c]

Show more

https://doi.org/10.1016/j.artint.2020.103348

Get rights and content

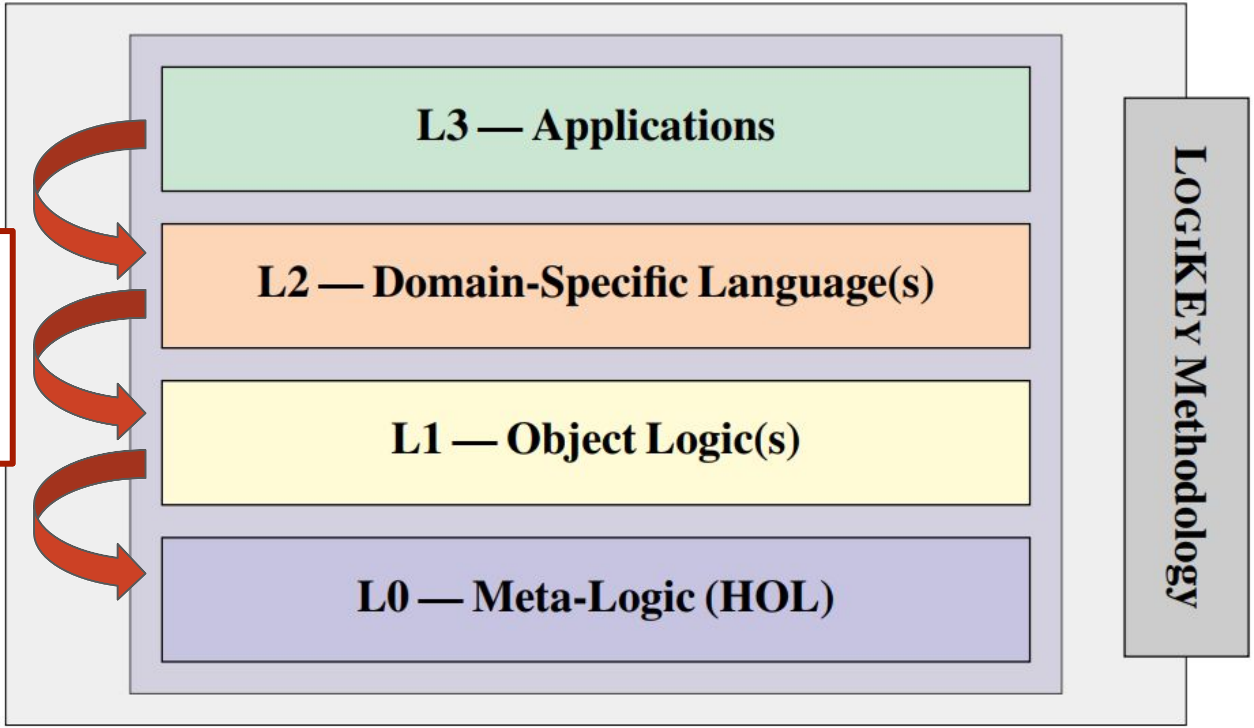
## LogiKEy:
## Flexible Ethico-Legal Reasoning (in HOL)



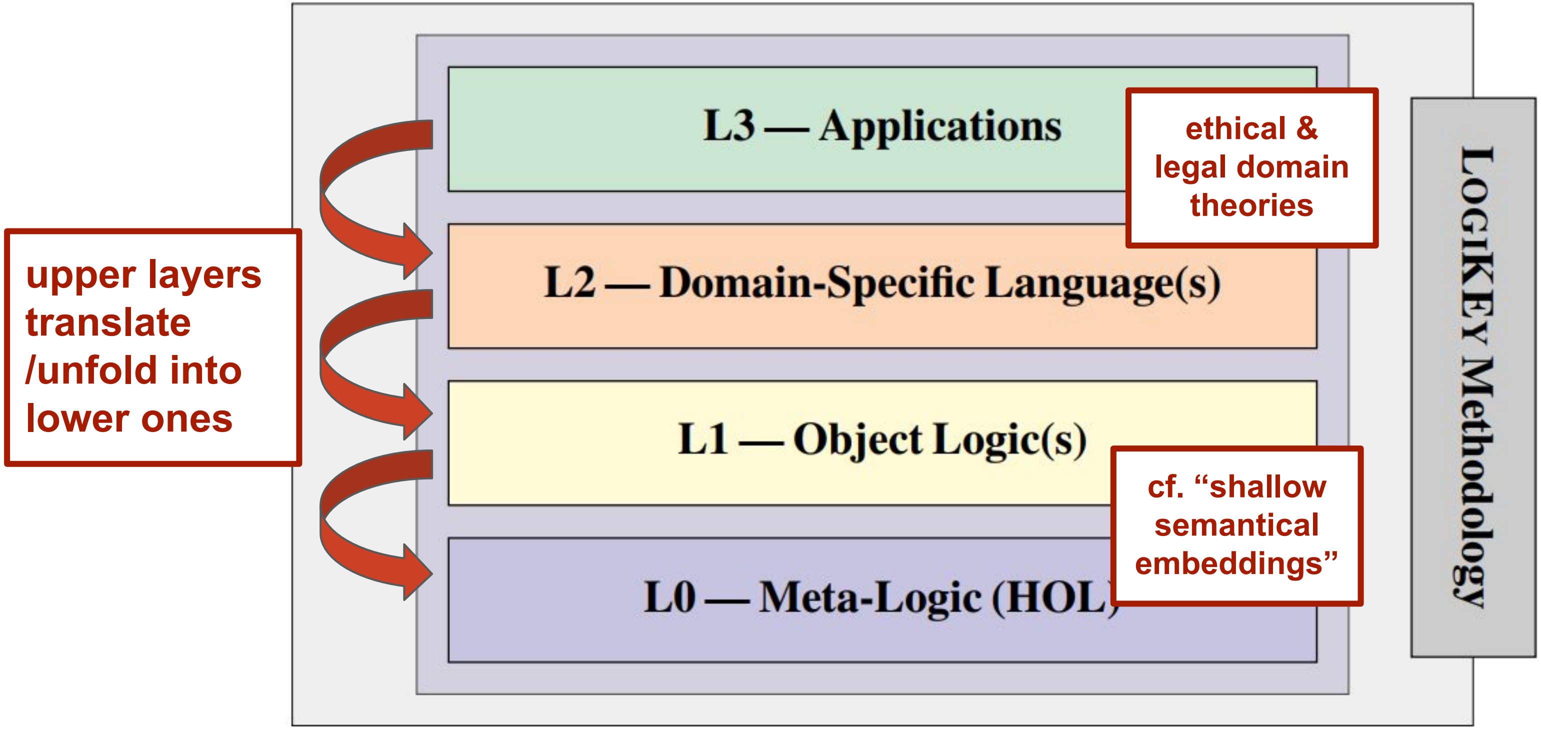


# A Digression: The LogiKEy Layers

upper layers translate /unfold into lower ones

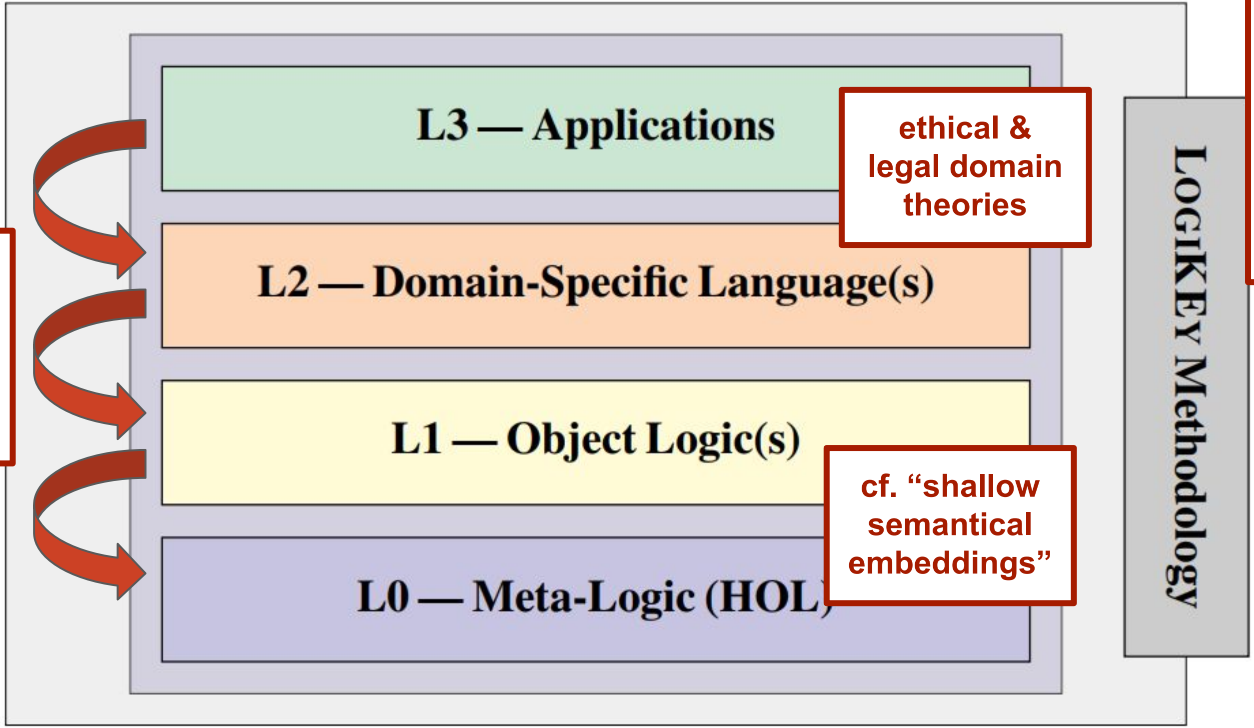


# A Digression: The LogiKEy Layers



**A Digression:**

# The LogiKEy Layers

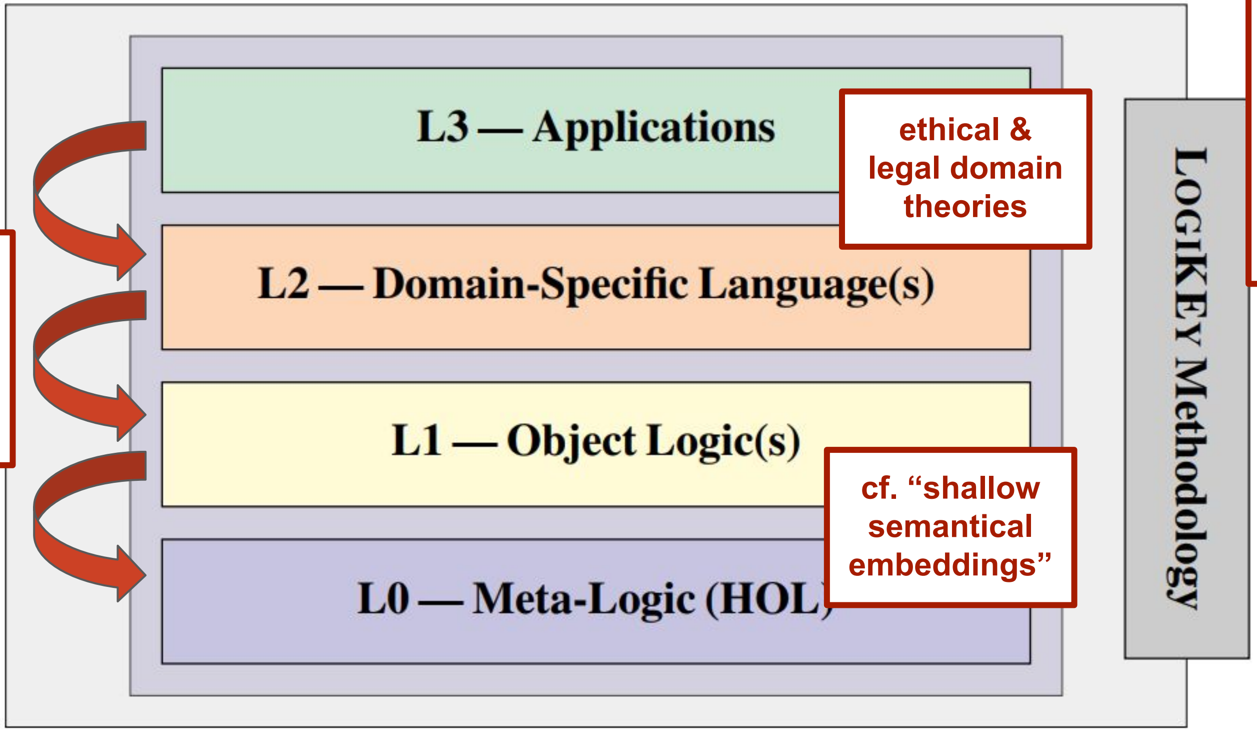



upper layers translate /unfold into lower ones

Different roles:

- Domain Expert
- Knw. Engineer
- Logic Engineer
- Programmer
- more…

# A Digression: The LogiKEy Layers




upper layers translate /unfold into lower ones

ethical & legal domain theories

cf. “shallow semantical embeddings”

**Different roles:**

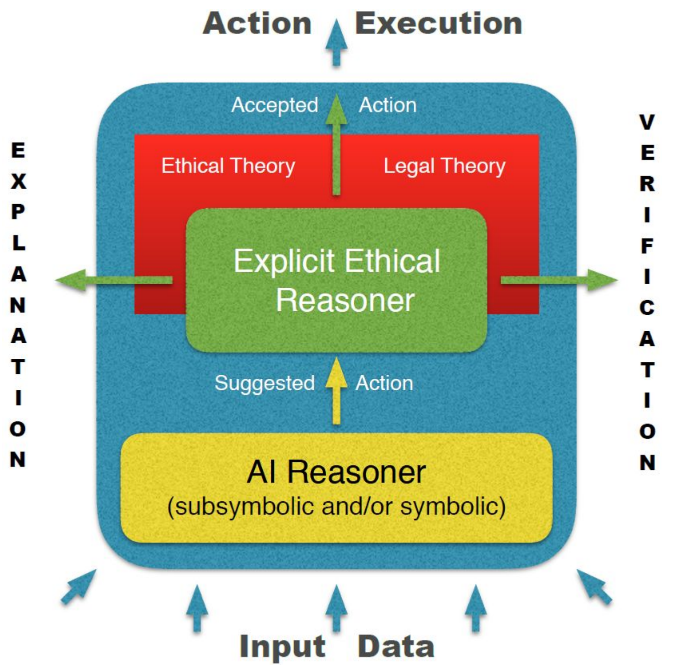
- Domain Expert
- Knw. Engineer
- Logic Engineer
- Programmer
- more…

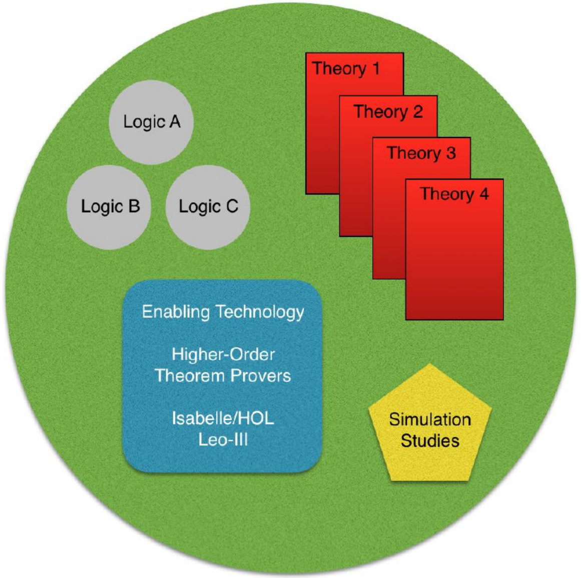
**Division of Labour!**

# A Digression:



## LogiKEy as a Framework for Trustworthy AI



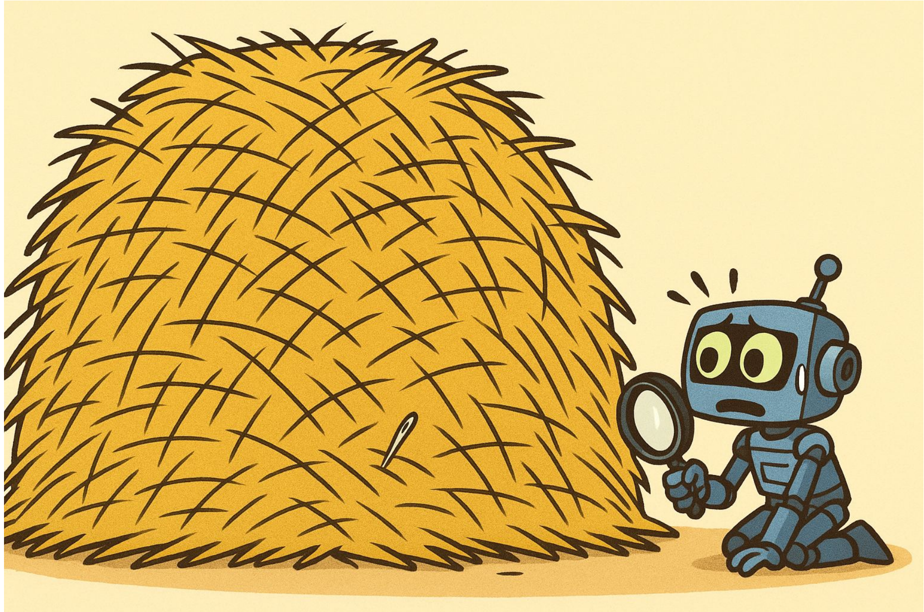




Benzmüller, Parent & van der Torre. “Designing Normative Theories of Ethical Reasoning: Formal Framework, Methodology, and Tool Support”. Artificial Intelligence (2020)

# “Layered” Reasoning

Automated Theorem Proving is hard…

…like searching for a needle in a haystack



# Automated Theorem Proving is hard…

## …or even harder!

## The Space of Proofs

***"… The overriding difficulty met at every turn was the unimaginably vast size of the space of proofs, a space in which all proofs solving a particular problem at hand might well be as unreachable as the farthest stars in the most distant galaxies. Consideration of quite short proofs suffices to illustrate this combinatorial explosion: even for systems of logic of the sort studied in this book that have just one axiom, for instance, there can be more 10-step proofs than kilometers in a light year, more 15-step proofs than stars in a trillion Milky Ways."***

**Foreword (by Dolph Ulrich) of the book "Automated Reasoning and the Discovery of Missing and Elegant Proofs" by Larry Wos & Gail W. Pieper**

# Automated Theorem Proving is hard…

… that came from an automated reasoning in first-order logic (FOL) book

# Automated Theorem Proving is hard…

… that came from an automated reasoning in first-order logic (FOL) book

What about HOL?

# Automated Theorem Proving is hard…

… that came from an automated reasoning in first-order logic (FOL) book

What about HOL?

- Worst-case theoretical analysis:

# Automated Theorem Proving is hard…

… that came from an automated reasoning in first-order logic (FOL) book

What about HOL?

- Worst-case theoretical analysis: **“intractable”**

# Automated Theorem Proving is hard…

… that came from an automated reasoning in first-order logic (FOL) book

What about HOL?

- Worst-case theoretical analysis:
  **"intractable"**
- Software engineering/AI:

# Automated Theorem Proving is hard…

… that came from an automated reasoning in first-order logic (FOL) book

What about HOL?

- Worst-case theoretical analysis:
  **“intractable”**
- Software engineering/AI:
  **“it depends”**

# Automated Theorem Proving is hard…

… that came from an automated reasoning in first-order logic (FOL) book

What about HOL?

- Worst-case theoretical analysis:
  **“intractable”**
- Software engineering/AI:
  **“it depends”**

**Domain specific!**

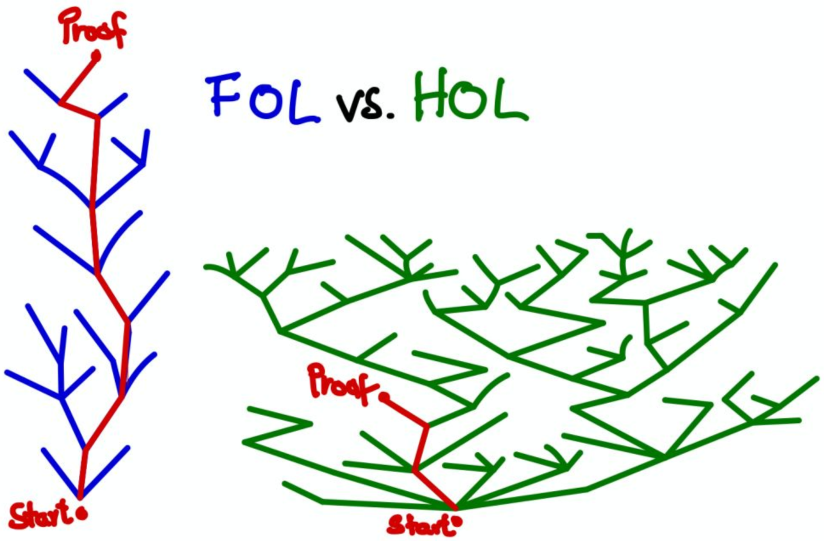
# Automated Theorem Proving is hard…

… that came from an automated reasoning in first-order logic (FOL) book

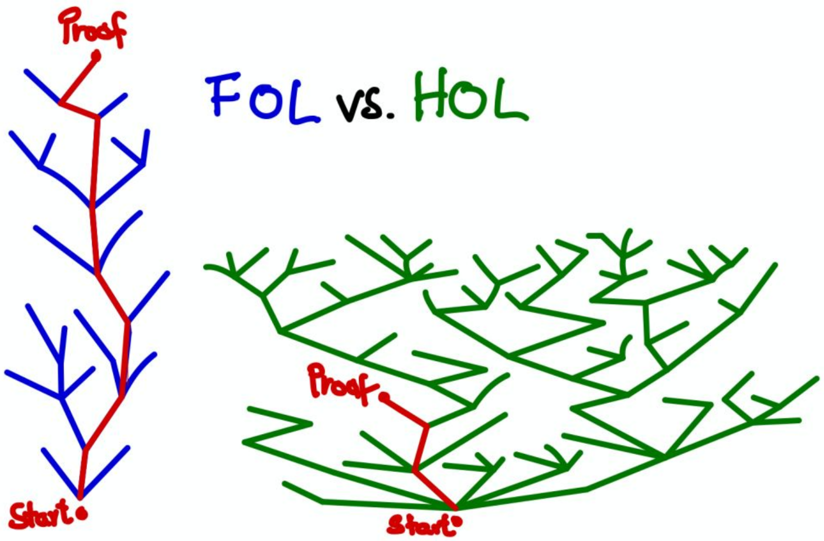
What about HOL?

- **Optimistic** theoretical analysis:

# Automated Theorem Proving is hard…

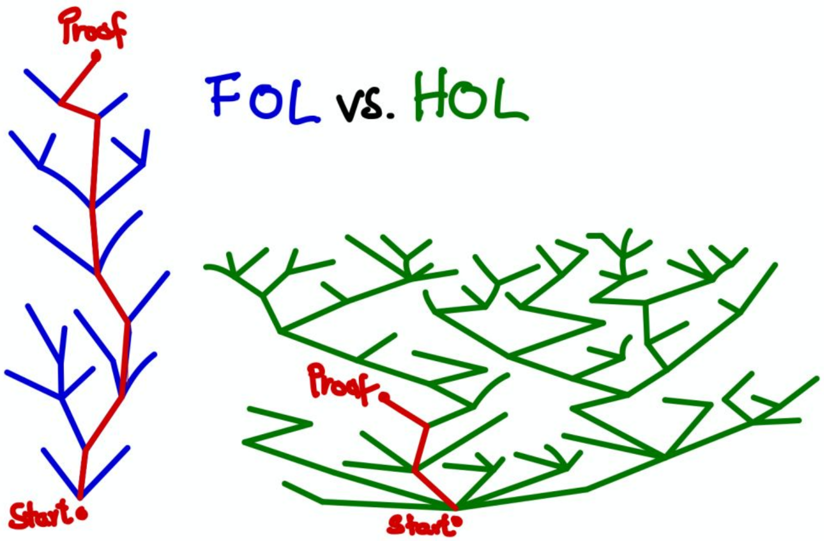


# Automated Theorem Proving is hard…




Automated reasoning in HOL is more complex, but more rewarding!

# Automated Theorem Proving is hard…




Proofs in HOL can be short, elegant (and arguably more intuitive)

# A Digression:

## Wormholes in Proof-Space

It is possible to obtain (hyper-)exponentially smaller proofs to a given problem by moving from an N-order encoding to an to N+1-order one.

Classic references:

- **(claim)** K. Gödel *“Über die Länge von Beweisen”* (1936)
- **(proof)** S. Buss *“On Gödel's theorems on lengths of proofs. I-II”* (1994-95)

# A Digression:

# Wormholes in Proof-Space

**Paper advertisement:**

**"*Who Finds the Short Proof?*"** (Benzmüller, Fuenmayor, Steen & Sutcliffe, 2022)

Follow-up for:

*"A Lost Proof"* (Benzmüller & Kerber, 2001)

Motivated by:

*"A Curious Inference"* (Boolos 1987)

*"Don't eliminate cut!"* (Boolos 1984)

# A Digression:

# Wormholes in Proof-Space

**Who Finds the Short Proof?**
Folbert and Holly (waiting at the gates of heaven) become engaged in a **theorem proving contest** in which they have to pose **first-order** proof problems to each other, and the one whose **ATP solves the given problem the faster will be admitted** to heaven. **Folbert goes for first-order ATPs and Holly for higher-order ATPs.**

# A Digression:

# Wormholes in Proof-Space

**Who Finds the Short Proof?**
Folbert and Holly (waiting at the gates of heaven) become engaged in a **theorem proving contest** in which they have to pose **first-order** proof problems to each other, and the one whose **ATP solves the given problem the faster will be admitted** to heaven. **Folbert goes for first-order ATPs** and **Holly for higher-order ATPs**.

We **quote** from Benzmüller, Fuenmayor, Steen & Sutcliffe (2022):

*"**Key to Holly's advantage are the (hyper-)exponentially shorter proofs** that are possible as one moves up the ladder of expressiveness from first-order logic to second-order logic, to third-order logic, and so on [Gö36]. **The fact that the proof problems are stated in FO logic does not matter**. When stating the same problem in the same FO way but in higher-order logic, much shorter proofs are possible, some of which might even be (hyper-)exponentially shorter than the proofs that can be found with comparatively inexpressive FO ATPs. A very prominent example of such a short proof is that of **Boolos' Curious Inference [Boo87]**."*

A Digression:

# Wormholes in Proof-Space

**Boolos "A Curious Inference" (1987):**

$$\forall n. f(n, e) = s(e) \tag{A1}$$

$$\forall y. f(e, s(y)) = s(s(f(e, y))) \tag{A2}$$

$$\forall x\, y. f(s(x), s(y)) = f(x, f(s(x), y)) \tag{A3}$$

$$d(e) \tag{A4}$$

$$\forall x. d(x) \rightarrow d(s(x)) \tag{A5}$$

$$d(f(s(s(s(s(e)))), s(s(s(s(e)))))) \tag{C}$$

# A Digression:

# Wormholes in Proof-Space

$$\forall n. f(n, e) = s(e) \quad \text{(A1)}$$

$$\forall y. f(e, s(y)) = s(s(f(e, y))) \quad \text{(A2)}$$

$$\forall x\, y. f(s(x), s(y)) = f(x, f(s(x), y)) \quad \text{(A3)}$$

$$d(e) \quad \text{(A4)}$$

$$\forall x. d(x) \rightarrow d(s(x)) \quad \text{(A5)}$$

$$d(f(s(s(s(s(e)))), s(s(s(s(e)))))) \quad \text{(C)}$$

- Axioms **A1-A3** capture the fact that *f* belongs to a class of extremely fast growing functions, also known as Ackermann(-style) functions.

# A Digression:

# Wormholes in Proof-Space

$$\forall n. f(n, e) = s(e) \quad \text{(A1)}$$

$$\forall y. f(e, s(y)) = s(s(f(e, y))) \quad \text{(A2)}$$

$$\forall x\, y. f(s(x), s(y)) = f(x, f(s(x), y)) \quad \text{(A3)}$$

$$d(e) \quad \text{(A4)}$$

$$\forall x. d(x) \rightarrow d(s(x)) \quad \text{(A5)}$$

$$d(f(s(s(s(s(e)))), s(s(s(s(e)))))) \quad \text{(C)}$$

- Axioms **A1-A3** capture the fact that *f* belongs to a class of extremely fast growing functions, also known as Ackermann(-style) functions.
- Axioms **A4-A5** introduce an inductive set “**d**”

# A Digression:

## Wormholes in Proof-Space

$$\forall n. f(n, e) = s(e) \quad \text{(A1)}$$

$$\forall y. f(e, s(y)) = s(s(f(e, y))) \quad \text{(A2)}$$

$$\forall x\, y. f(s(x), s(y)) = f(x, f(s(x), y)) \quad \text{(A3)}$$

$$d(e) \quad \text{(A4)}$$

$$\forall x. d(x) \rightarrow d(s(x)) \quad \text{(A5)}$$

$$d(f(s(s(s(s(e)))), s(s(s(s(e)))))) \quad \text{(C)}$$

- Exhaustive evaluation of the term ***f(s(s(s(s(e)))), s(s(s(s(e)))))*** with the recursive equations **A1-A3** unfolds it to a term that contains more "*s*" than there are atoms in the universe!

# A Digression:

## Wormholes in Proof-Space

$$\forall n. f(n, e) = s(e) \qquad \text{(A1)}$$

$$\forall y. f(e, s(y)) = s(s(f(e, y))) \qquad \text{(A2)}$$

$$\forall x\, y. f(s(x), s(y)) = f(x, f(s(x), y)) \qquad \text{(A3)}$$

$$d(e) \qquad \text{(A4)}$$

$$\forall x. d(x) \rightarrow d(s(x)) \qquad \text{(A5)}$$

$$d(f(s(s(s(s(e)))), s(s(s(s(e)))))) \qquad \text{(C)}$$

- Exhaustive evaluation of the term ***f(s(s(s(s(e)))), s(s(s(s(e)))))*** with the recursive equations **A1-A3** unfolds it to a term that contains more “*s*” than there are atoms in the universe!
- This proof problem is solvable in a “cut-free” first-order calculus by applying an astronomically large number of modus ponens steps to **A4** and (instances of) **A5**.

# A Digression:

# Wormholes in Proof-Space

$$\forall n. f(n, e) = s(e) \tag{A1}$$

$$\forall y. f(e, s(y)) = s(s(f(e, y))) \tag{A2}$$

$$\forall x\, y. f(s(x), s(y)) = f(x, f(s(x), y)) \tag{A3}$$

$$d(e) \tag{A4}$$

$$\forall x. d(x) \rightarrow d(s(x)) \tag{A5}$$

$$d(f(s(s(s(s(e)))), s(s(s(s(e)))))) \tag{C}$$

- When given the following two definitions, various **higher-order ATPs can find a "one-page" proof**.

$$ind = \lambda X. Xe \wedge \forall x. Xx \rightarrow Xsx \tag{Def\_$ind$}$$

$$p = \lambda x\, y. (\lambda z. \forall X.\ ind\ X \rightarrow Xz)\ fxy \tag{Def\_$p$}$$

# A Digression:

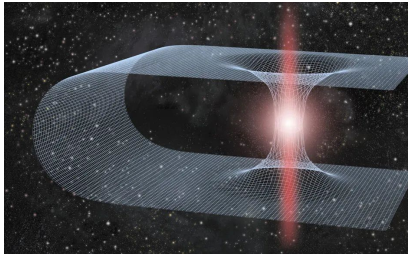
# Wormholes in Proof-Space

$$\forall n. f(n, e) = s(e) \quad \text{(A1)}$$
$$\forall y. f(e, s(y)) = s(s(f(e, y))) \quad \text{(A2)}$$
$$\forall x\, y. f(s(x), s(y)) = f(x, f(s(x), y)) \quad \text{(A3)}$$
$$d(e) \quad \text{(A4)}$$
$$\forall x. d(x) \rightarrow d(s(x)) \quad \text{(A5)}$$

$$d(f(s(s(s(s(e)))), s(s(s(s(e)))))) \quad \text{(C)}$$

- When given the following two definitions, various **higher-order ATPs can find a “one-page” proof**.

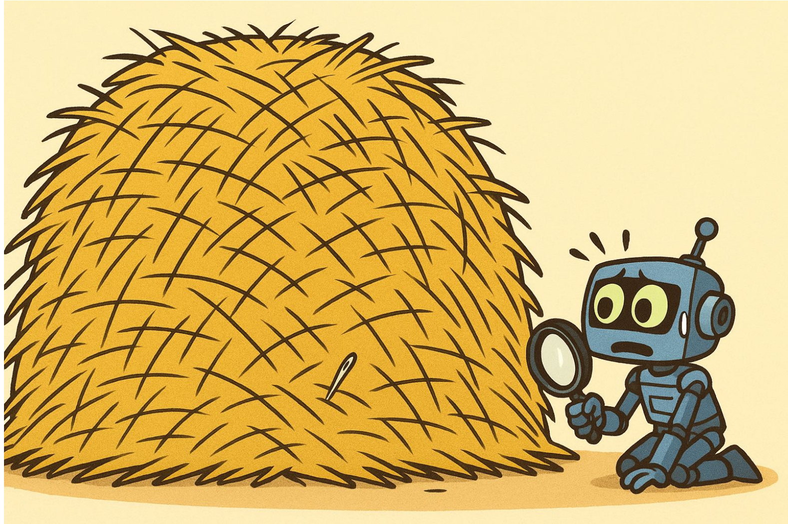
$$ind = \lambda X.\, Xe \wedge \forall x.\, Xx \rightarrow Xsx \quad (\text{Def\_}ind)$$
$$p = \lambda x\, y.\, (\lambda z. \forall X.\ ind\ X \rightarrow Xz)\ fxy \quad (\text{Def\_}p)$$



# Automated Theorem Proving is hard…

## …like searching for a needle in a haystack

Where does the haystack come from?

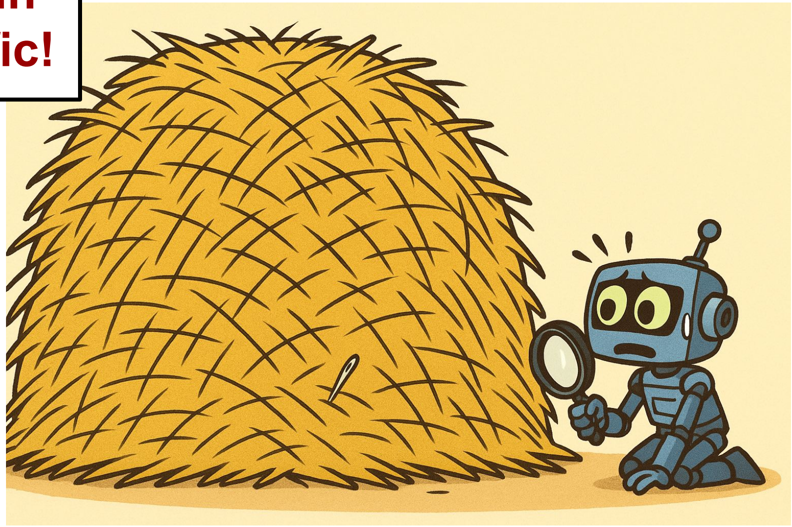


# Automated Theorem Proving is hard…

…like searching for a needle in a haystack

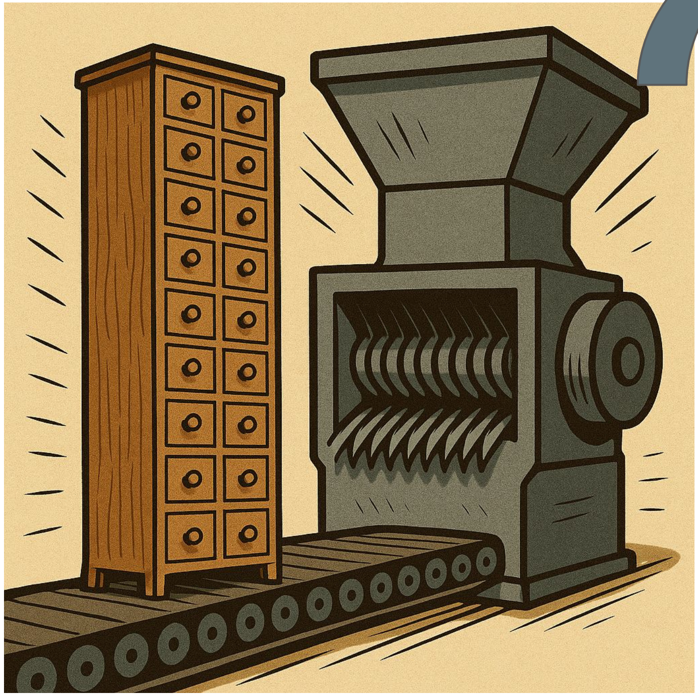
Where does the haystack come from?

**Domain specific!**

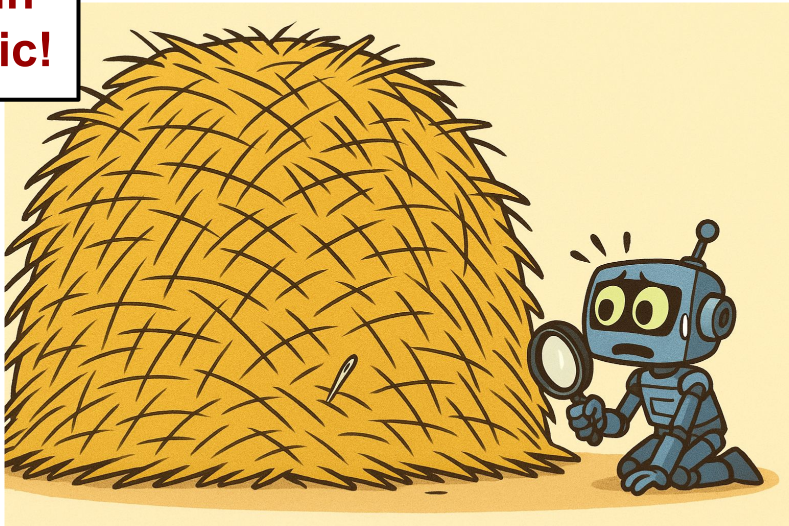


# Automated Theorem Proving is hard…

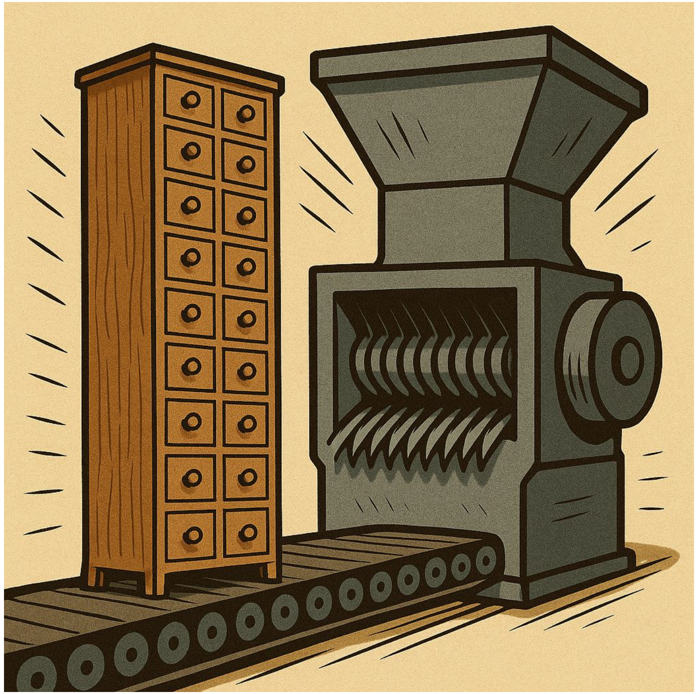
## …like searching for a needle in a haystack



**Domain specific!**



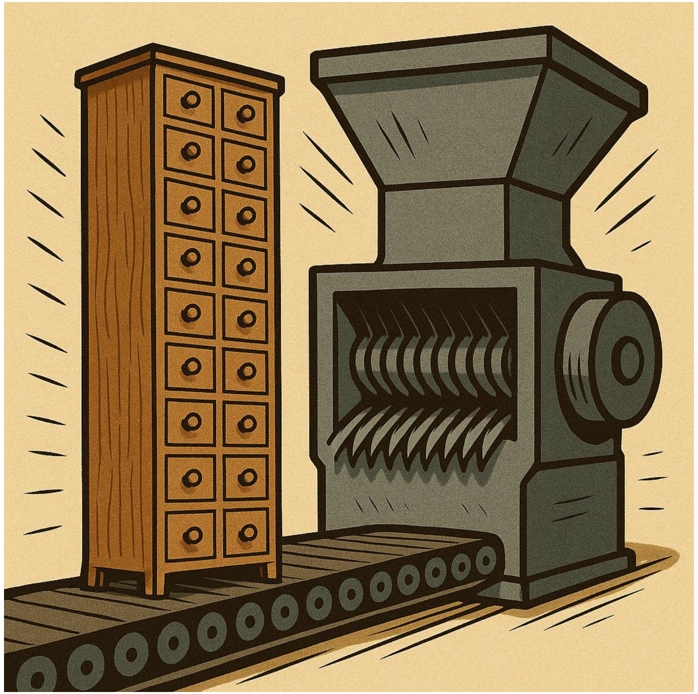
# The “Problem of Formalization”



➔ Conversion to “normal forms”

CNF, DNF, etc.

# The “Problem of Formalization”

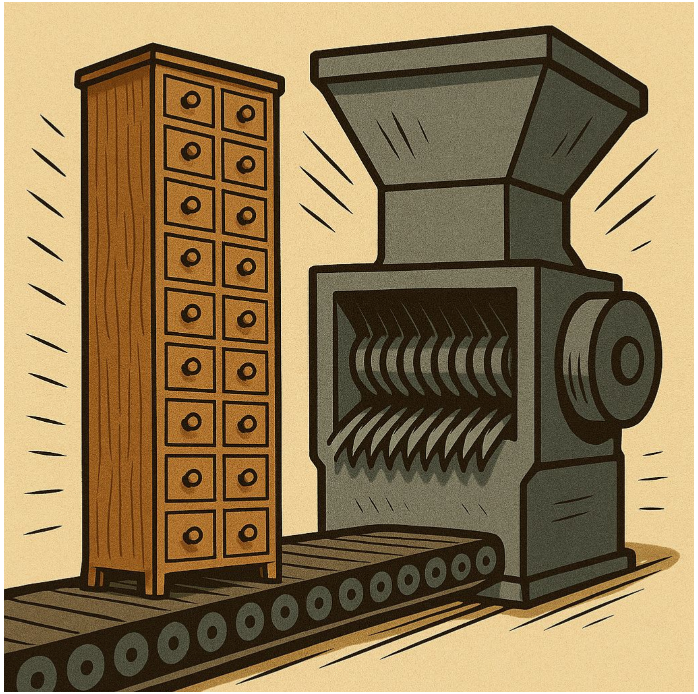


- ➔ Conversion to “normal forms”

  CNF, DNF, etc.

- ➔ Lambda reduction (beta-, eta-, etc.)

# The “Problem of Formalization”



- ➔ Conversion to “normal forms”
  - CNF, DNF, etc.
- ➔ Lambda reduction (beta-, eta-, etc.)
- ➔ **Definition expansion and contraction**

# Another digression:
# “The Problem of Definition Expansion and Contraction”

- “Research Problem #30” in Larry Wo’s list of 33 open research problems in automated reasoning (1987)

# Another digression:
# “The Problem of Definition Expansion and Contraction”

- “Research Problem #30” in Larry Wo’s list of 33 open research problems in automated reasoning (1987)
- Subject of intense work in the 80’s (for first-order reasoning):
    - “Peeking” (UT theorem provers by Bledsoe & co.)
    - “Gazing” (GAZER prover by Barker-Plummer & co.)
    - Abstraction-based proving (Plaisted “Abstraction mappings in mechanical theorem proving” 1980; Giunchiglia & Walsh “Abstract Theorem Proving” 1989-1992)

# Another digression:
# "The Problem of Definition Expansion and Contraction"

- "Research Problem #30" in Larry Wo's list of 33 open research problems in automated reasoning (1987)
- Subject of intense work in the 80's (for first-order reasoning):
  - "Peeking" (UT theorem provers by Bledsoe & co.)
  - "Gazing" (GAZER prover by Barker-Plummer & co.)
  - Abstraction-based proving (Plaisted "Abstraction mappings in mechanical theorem proving" 1980; Giunchiglia & Walsh "Abstract Theorem Proving" 1989-1992)
- Less references towards end of 90's. Some for higher-order proving are:
  - "Dual instantiation" (TPS prover, cf. Bishop & Andrews' "Selectively Instantiating Definitions" 1998)
  - "Proof planning" (Bundy & co; cf. also Omega system)

# Another digression: “The Problem of Definition Expansion and Contraction”

- “Research Problem #30” in Larry Wo’s list of 33 open research problems in automated reasoning (1987)
- Subject of intense work in the 80’s (for first-order reasoning):
    - “Peeking” (UT theorem provers by Bledsoe & co.)
    - “Gazing” (GAZER prover by Barker-Plummer & co.)
    - Abstraction-based proving (Plaisted “Abstraction mappings in mechanical theorem proving” 1980; Giunchiglia & Walsh “Abstract Theorem Proving” 1989-1992)
- Less references towards end of 90’s. Some for higher-order proving are:
    - “Dual instantiation” (TPS prover, cf. Bishop & Andrews’ “Selectively Instantiating Definitions” 1998)
    - “Proof planning” (Bundy & co; cf. also Omega system)
- Almost nothing afterwards in ATP. Some hints in ITP (premise selection, “hammers”).

# Another digression:
# “The Problem of Definition Expansion and Contraction”

**Definitions should become first-class citizens in ATP!**

- They carry important (domain-specific) proof-relevant information.

# Another digression:
# “The Problem of Definition Expansion and Contraction”

**Definitions should become first-class citizens in ATP!**

- They carry important (domain-specific) proof-relevant information.
- Don’t unfold them away at the beginning! Rather: selective instantiation during proof search (cf. TPS “dual instantiation”)

# Another digression:
# “The Problem of Definition Expansion and Contraction”

**Definitions should become first-class citizens in ATP!**

- They carry important (domain-specific) proof-relevant information.
- Don’t unfold them away at the beginning! Rather: selective instantiation during proof search (cf. TPS “dual instantiation”)
- If not provided by the user:

# Another digression:
# "The Problem of Definition Expansion and Contraction"

**Definitions should become first-class citizens in ATP!**

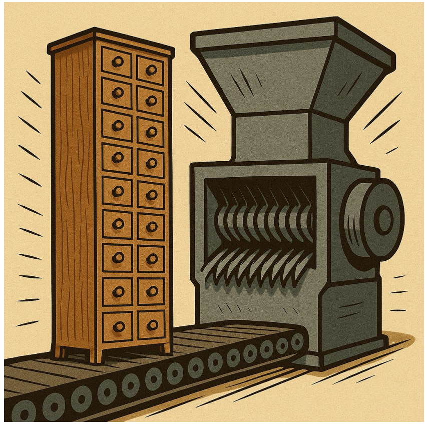
- They carry important (domain-specific) proof-relevant information.
- Don't unfold them away at the beginning! Rather: selective instantiation during proof search (cf. TPS "dual instantiation")
- If not provided by the user: discover them!

# Another digression:
# "The Problem of Definition Expansion and Contraction"

**Definitions should become first-class citizens in ATP!**

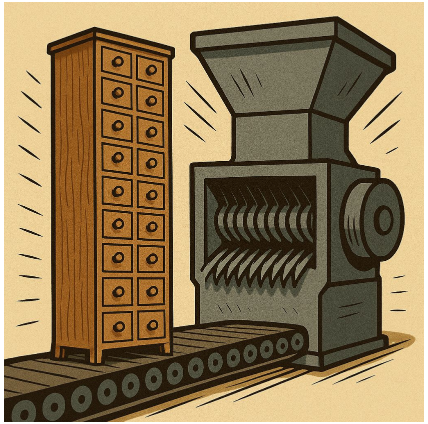
- They carry important (domain-specific) proof-relevant information.
- Don't unfold them away at the beginning! Rather: selective instantiation during proof search (cf. TPS "dual instantiation")
- If not provided by the user: discover them!
- Fusion of horizons: mixing **"top-down"** & **"bottom-up"**:
  - **Top-down** proof planning (Bundy & co.; cf. also OMEGA system team)
  - **Bottom-up** theory construction (Buchberger's *Theorema* system; systems like *IsaScheme*, *IsaCosy*, *Hipster*, etc.)

# The “Problem of Formalization”



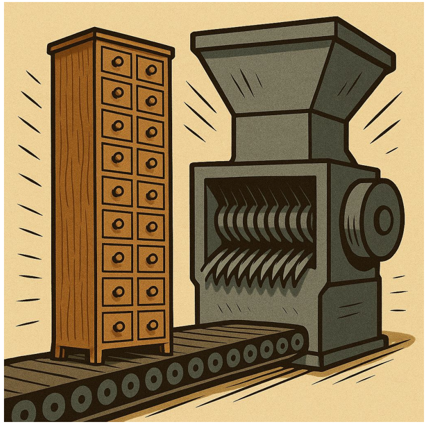
- Formalization is an essential part of proving!

# The “Problem of Formalization”



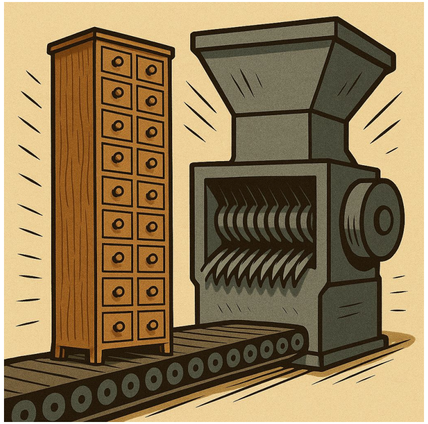
- Formalization is an essential part of proving!
- Previous work on “computational hermeneutics”
  - *Slogan: “every formalization is an interpretation”*

# The “Problem of Formalization”



- Formalization is an essential part of proving!
- Previous work on “computational hermeneutics”
  - *Slogan: “every formalization is an interpretation”*
  - Holistic approach where formalization and proving (as an instance of argumentation) are two sides of the same coin.

# The "Problem of Formalization"



- Formalization is an essential part of proving!
- Previous work on "computational hermeneutics"
  - *Slogan: "every formalization is an interpretation"*
  - Holistic approach where formalization and proving (as an instance of argumentation) are two sides of the same coin.

Cf. Fuenmayor & Benzmüller (2019)

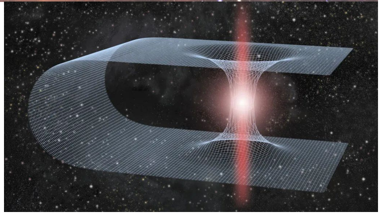
- *"A computational-hermeneutic approach for conceptual explicitation"*
- *"Computational hermeneutics: An integrated approach for the logical analysis of natural-language arguments"*

# The “Problem of Formalization”

- “Proof-space” → “interpretation-space”

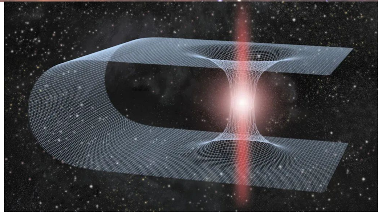
# The “Problem of Formalization”

- “Proof-space” → “interpretation-space”



- Proof search → definition search (theory construction)

# The “Problem of Formalization”



- “Proof-space” → “interpretation-space”

- Proof search → definition search (theory construction)

- **Next step:** A combinators-based (aka. “point-free”) mathematical language (on top of HOL) as a vehicle to navigate the interpretation space.

# The “Building Blocks” Approach to Mathematical Logic

- ➢ First presented in a 1920 talk by Moses Schönfinkel

# The “Building Blocks” Approach to Mathematical Logic

- ➢ First presented in a 1920 talk by Moses Schönfinkel
- ➢ Schönfinkel “disappeared” shortly after.

# The “Building Blocks” Approach to Mathematical Logic

- ➢ First presented in a 1920 talk by Moses Schönfinkel

- ➢ Schönfinkel “disappeared” shortly after.

- ➢ Published in 1924 as *“Über die Bausteine der mathematischen Logik”* in the *Mathematische Annalen*

# The “Building Blocks” Approach to Mathematical Logic

- ➢ First presented in a 1920 talk by Moses Schönfinkel
- ➢ Schönfinkel “disappeared” shortly after.
- ➢ Published in 1924 as *“Über die Bausteine der mathematischen Logik”* in the *Mathematische Annalen*
- ➢ Stephen Wolfram has recently done some research on what (may have) happened.

# The “Building Blocks” Approach to Mathematical Logic

Recalling:

HOL = *Simply-Typed Lambda Calculus extended with a generic constant symbol denoting (dis)equality*

# The “Building Blocks” Approach to Universal/Pluralistic Logic(s)

My work: The best of both worlds!

# The “Building Blocks” Approach to Universal/Pluralistic Logic(s)

My work: The best of both worlds!

Schönfinkel’s “building-blocks” (aka. Curry’s “combinators”)

# The “Building Blocks” Approach to Universal/Pluralistic Logic(s)

My work: The best of both worlds!

Schönfinkel’s “building-blocks” (aka. Curry’s “combinators”)

on top of

Simply-Typed Lambda Calculus + (dis)equality (aka. “HOL”)

# The “Building Blocks” Approach to Universal/Pluralistic Logic(s)

My work: The best of both worlds!

Schönfinkel’s “building-blocks” (aka. Curry’s “combinators”)

on top of

Simply-Typed Lambda Calculus + (dis)equality (aka. “HOL”)

\+ (shallow) embeddings of non-classical logics (and DSLs)

# The “Building Blocks” Approach to Universal/Pluralistic Logic(s)

My work: The best of both worlds!

Schönfinkel’s “building-blocks” (aka. Curry’s “combinators”)

on top of

Simply-Typed Lambda Calculus + (dis)equality (aka. “HOL”)

\+ (shallow) embeddings of non-classical logics (and DSLs)

An implementation: “Combinatory Logic Bricks Library”

https://github.com/davfuenmayor/logic-bricks

# The “Building Blocks” Approach to Universal/Pluralistic Logic(s)

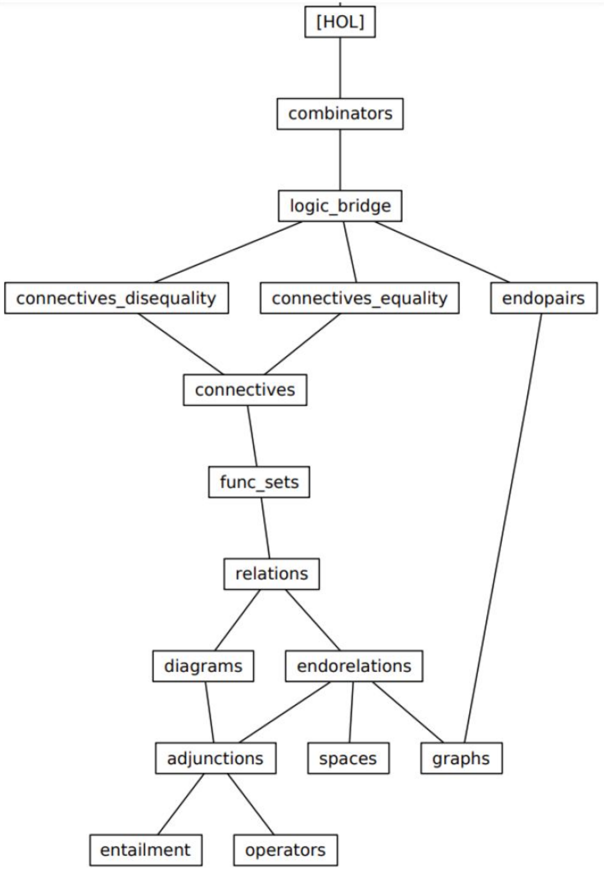
A very quick “one-minute” demo:

**Combinatory Logic Bricks Library**

https://github.com/davfuenmayor/logic-bricks

# The Combinatory Logic Bricks Library

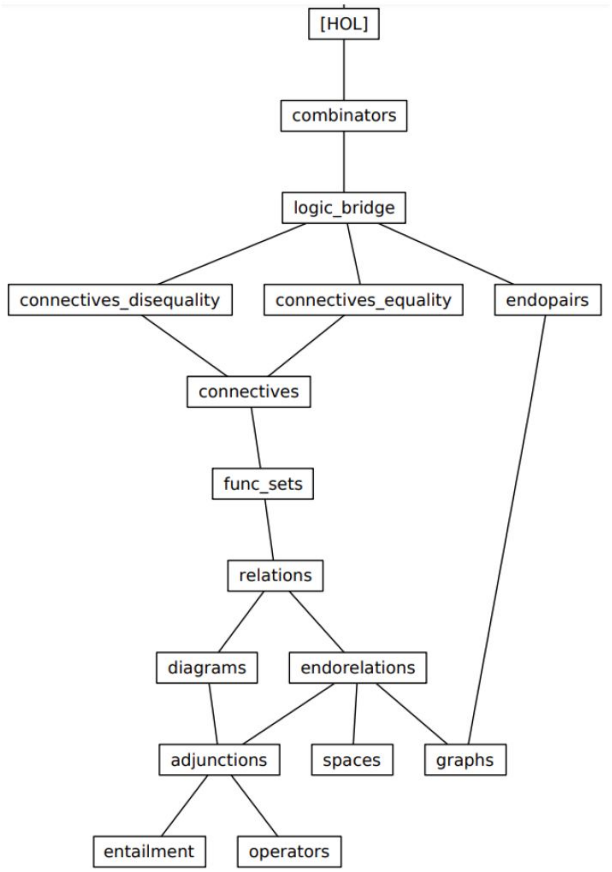
➢ Result of several years of theory exploration in the context of non-classical logics (cf. LogiKEy)



# The Combinatory Logic Bricks Library

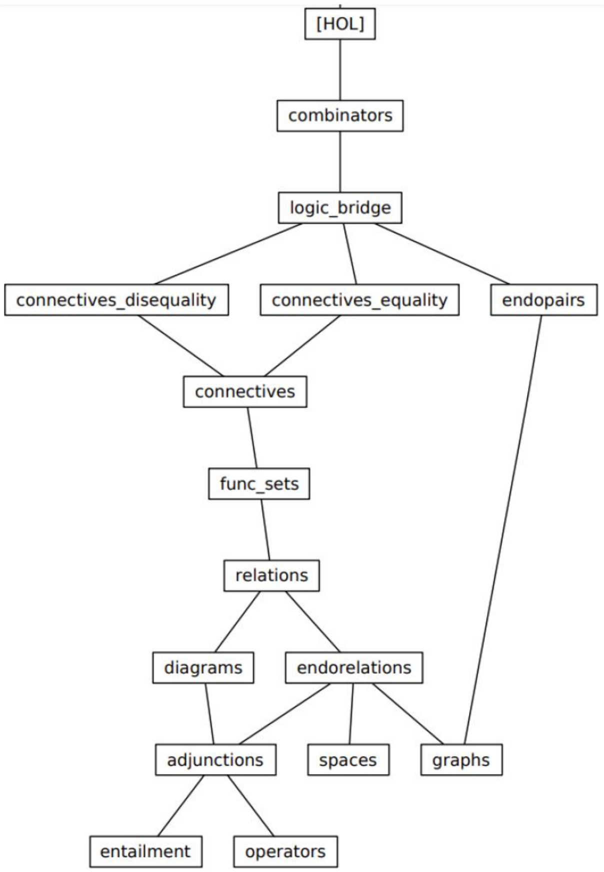
➢ Result of several years of theory exploration in the context of non-classical logics (cf. LogiKEy)

➢ Based upon a theory of “multidimensional combinators” as basic functional “plumbing”.



# The Combinatory Logic Bricks Library

- ➢ Result of several years of theory exploration in the context of non-classical logics (cf. LogiKEy)
- ➢ Based upon a theory of “multidimensional combinators” as basic functional “plumbing”.
- ➢ Concepts are introduced as clusters of equivalent definitions (with the main one being “point-free”).



Yet another digression:

## Knowledge Representation and Reasoning without variables (aka. “point-free style”)

- ➢ The “compositionality mindset” taken to its extreme.

Yet another digression:

# Knowledge Representation and Reasoning without variables (aka. “point-free style”)

- ➢ The “compositionality mindset” taken to its extreme.
- ➢ Facilitates code generation and optimization, program synthesis, etc.

Yet another digression:

# Knowledge Representation and Reasoning without variables (aka. “point-free style”)

- The “compositionality mindset” taken to its extreme.
- Facilitates code generation and optimization, program synthesis, etc.
- Powerful technique in compilation (and optimization) of functional programming languages (cf. D. Turner’s seminal *“A New Implementation Technique for Applicative Languages”* 1979)

Yet another digression:

## Knowledge Representation and Reasoning without variables (aka. "point-free style")

- The "compositionality mindset" taken to its extreme.
- Facilitates code generation and optimization, program synthesis, etc.
- Powerful technique in compilation (and optimization) of functional programming languages (cf. D. Turner's seminal *"A New Implementation Technique for Applicative Languages"* 1979)
- Lends itself to a "dataflow" programming paradigm (cf. VHDL, Verilog, APL, Linda, Lustre, TensorFlow, etc.)

Yet another digression:

## Knowledge Representation and Reasoning without variables (aka. “point-free style”)

- The “compositionality mindset” taken to its extreme.
- Facilitates code generation and optimization, program synthesis, etc.
- Powerful technique in compilation (and optimization) of functional programming languages (cf. D. Turner’s seminal *“A New Implementation Technique for Applicative Languages”* 1979)
- Lends itself to a “dataflow” programming paradigm (cf. VHDL, Verilog, APL, Linda, Lustre, TensorFlow, etc.)
- LLMs can cope with much easier (e.g. avoiding complex variable-substitution bookkeeping).

Yet another digression:

## Knowledge Representation and Reasoning without variables (aka. “point-free style”)

- ➢ Naturally allows for a “layered” theory construction paradigm (cf. *LogiKEy*).

Yet another digression:

## Knowledge Representation and Reasoning without variables (aka. “point-free style”)

- ➢ Naturally allows for a “layered” theory construction paradigm (cf. *LogiKEy*).
- ➢ Enables convenient selective definition expansion/contraction during proof search (e.g. “problem massaging” before sending it to a generalist prover for endgame).

Yet another digression:

## Knowledge Representation and Reasoning without variables (aka. “point-free style”)

- ➢ Naturally allows for a “layered” theory construction paradigm (cf. *LogiKEy*).
- ➢ Enables convenient selective definition expansion/contraction during proof search (e.g. “problem massaging” before sending it to a generalist prover for endgame).
- ➢ Makes explicit essential conceptual interrelations (cf. *computational hermeneutics*).

# Argumentative Reasoning Agents (ARAs)

- ➢ First intended "speakers" of this "universal mathematical language"

# Argumentative Reasoning Agents (ARAs)

- ➢ First intended “speakers” of this “universal mathematical language”
- ➢ Artificial

  as in “artificial intelligence”

# Argumentative Reasoning Agents (ARAs)

- First intended “speakers” of this “universal mathematical language”
- Artificial

  as in “artificial intelligence”
- Argumentative

  as in “argumentative discourse”

# Argumentative Reasoning Agents (ARAs)

- ➢ First intended “speakers” of this “universal mathematical language”
- ➢ Artificial

    as in “artificial intelligence”
- ➢ Argumentative

    as in “argumentative discourse”
- ➢ Reasoning

    as in “automated reasoning”

# Argumentative Reasoning Agents (ARAs)


- First intended "speakers" of this "universal mathematical language"
- Artificial

  as in "artificial intelligence"
- Argumentative

  as in "argumentative discourse"
- Reasoning

  as in "automated reasoning"
- Agents

  as in "multi-agent systems" but also "AI agents"

# A HOL-based Controlled Natural Language (CNL) for ARAs

- ARAs need to do math.

# A HOL-based Controlled Natural Language (CNL) for ARAs

- ARAs need to do math.
- ARAs need to understand (like) and communicate with humans.

# A HOL-based Controlled Natural Language (CNL) for ARAs

- ARAs need to do math.
- ARAs need to understand (like) and communicate with humans.
    - ARAs need to represent and analyse sentence structure.
    - ARAs need to properly handle modalities.
    - ARAs need to properly understand generalized quantifiers, indexicals and anaphora
    - …

# A HOL-based Controlled Natural Language (CNL) for ARAs

➢ ARAs need to create and communicate programs.

# A HOL-based Controlled Natural Language (CNL) for ARAs

- ➢ ARAs need to create and communicate programs.
- ➢ ARAs need to carry out planning and resource-sensitive reasoning.

# A HOL-based Controlled Natural Language (CNL) for ARAs

➢ ARAs need to create and communicate programs.

➢ ARAs need to carry out planning and resource-sensitive reasoning.

➢ ARAs need to engage in strategic reasoning and coalition building.

# A HOL-based Controlled Natural Language (CNL) for ARAs

- ARAs need to create and communicate programs.
- ARAs need to carry out planning and resource-sensitive reasoning.
- ARAs need to engage in strategic reasoning and coalition building.
- ARAs need to reason reliably with partial and contradictory evidence.

# A HOL-based Controlled Natural Language (CNL) for ARAs

- ARAs need to create and communicate programs.
- ARAs need to carry out planning and resource-sensitive reasoning.
- ARAs need to engage in strategic reasoning and coalition building.
- ARAs need to reason reliably with partial and contradictory evidence.
- ARAs need to engage in counterfactual reasoning.

# A HOL-based Controlled Natural Language (CNL) for ARAs

- ARAs need to create and communicate programs.
- ARAs need to carry out planning and resource-sensitive reasoning.
- ARAs need to engage in strategic reasoning and coalition building.
- ARAs need to reason reliably with partial and contradictory evidence.
- ARAs need to engage in counterfactual reasoning.
- ARAs need to represent and reason about argumentative discourse.

# A HOL-based Controlled Natural Language (CNL) for ARAs

The “Combinatory Logic Bricks” Isabelle/HOL library:

https://github.com/davfuenmayor/logic-bricks

is a first step towards implementing this CNL.

# ~~Past~~ Present Hype: Reasoning Agents

Contemporary AI is an 80’s party


# ~~Past~~ Present Hype: Reasoning Agents

Contemporary AI is an 80’s party

- Hot AI topics nowadays:
    - Reasoning
    - Planning
    - (Graph-)RAG (aka. KR)
    - “AI agents” (aka. MAS)

# ~~Past~~ Present Hype: Reasoning Agents

Contemporary AI is an 80’s party

- Hot AI topics nowadays:
  - Reasoning
  - Planning
  - (Graph-)RAG (aka. KR)
  - “AI agents” (aka. MAS)

- Are academic researchers in those topics attending?

# ~~Past~~ Present Hype: Reasoning Agents

LogiKEy is (in principle) good positioned to contribute:

# ~~Past~~ Present Hype: Reasoning Agents

LogiKEy is (in principle) good positioned to contribute:

Providing a **generalist** user interface (proof assistants, “HOL as universal metalanguage”, ATP “hammers”...)

# ~~Past~~ Present Hype: Reasoning Agents

LogiKEy is (in principle) good positioned to contribute:

Providing a **generalist** user interface (proof assistants, “HOL as universal metalanguage”, ATP “hammers”...)

Still catering to **domain-specificity** (NCLs, DSLs, external provers)

# ~~Past~~ Present Hype: Reasoning Agents

LogiKEy is (in principle) good positioned to contribute:

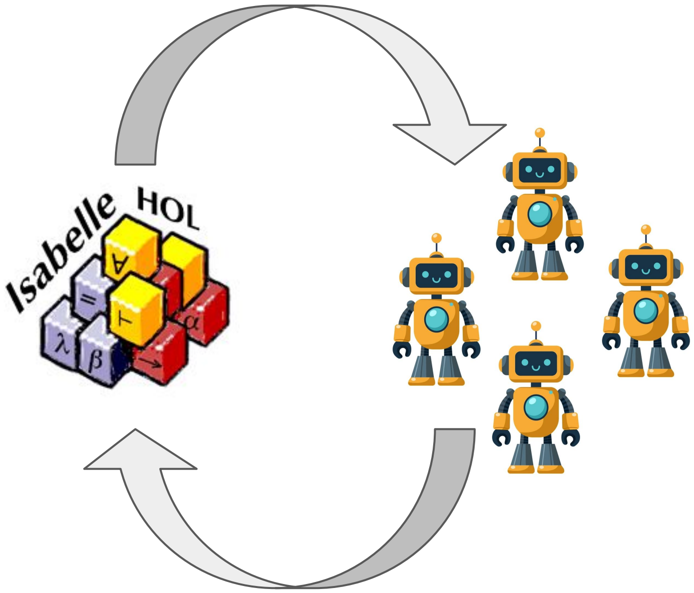
Providing a **generalist** user interface (proof assistants, “HOL as universal metalanguage”, ATP “hammers”...)

Still catering to **domain-specificity** (NCLs, DSLs, external provers)

Usability??

# Current Work: Truth-grounding signal for “AI Agents”

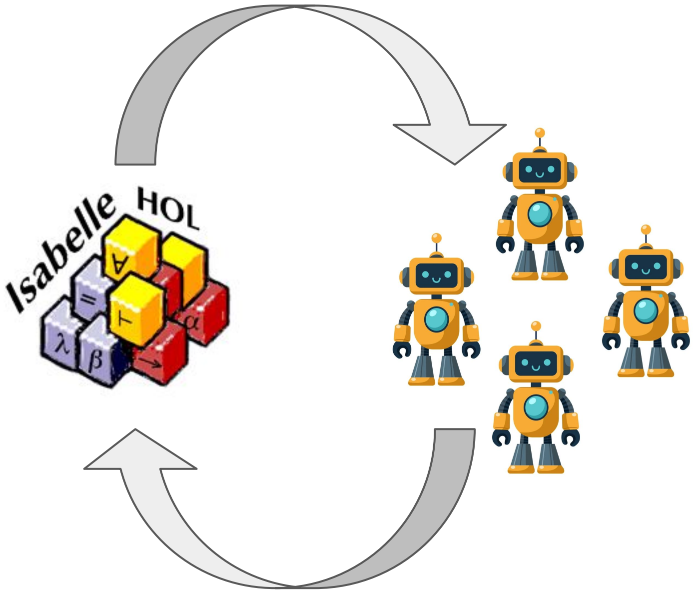
Integration of LLM-based agentic workflows (+RAG, +tool-calling, etc)



# Current Work: Truth-grounding signal for "AI Agents"

Integration of LLM-based agentic workflows (+RAG, +tool-calling, etc)

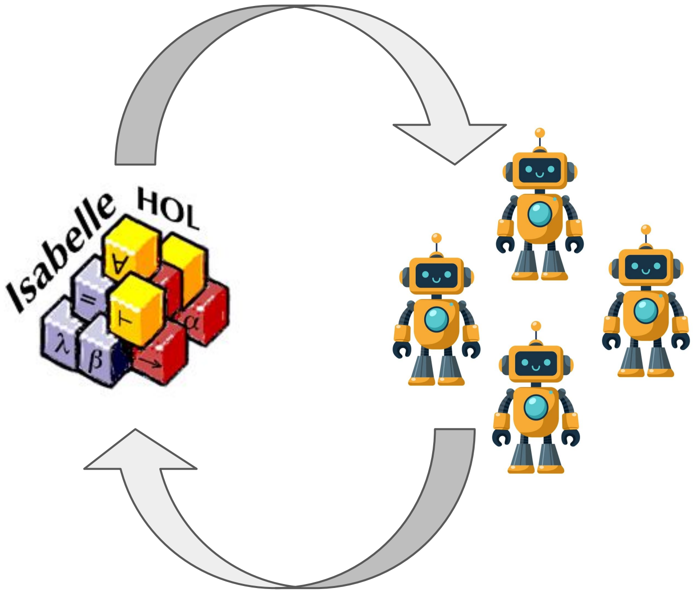
- AI-agents can generate/modify and formally check Isabelle theories (via Isabelle-server + Python/Elixir Isabelle-client)



# Current Work: Truth-grounding signal for "AI Agents"

Integration of LLM-based agentic workflows (+RAG, +tool-calling, etc)

- AI-agents can generate/modify and formally check Isabelle theories (via Isabelle-server + Python/Elixir Isabelle-client)
- Isabelle processes can invoke arbitrary external programs (e.g. LLM agents) as "abductive oracles" (e.g. to suggest "cut" definitions and lemmata)




# Thanks for your attention!

# Discussion / Q&A