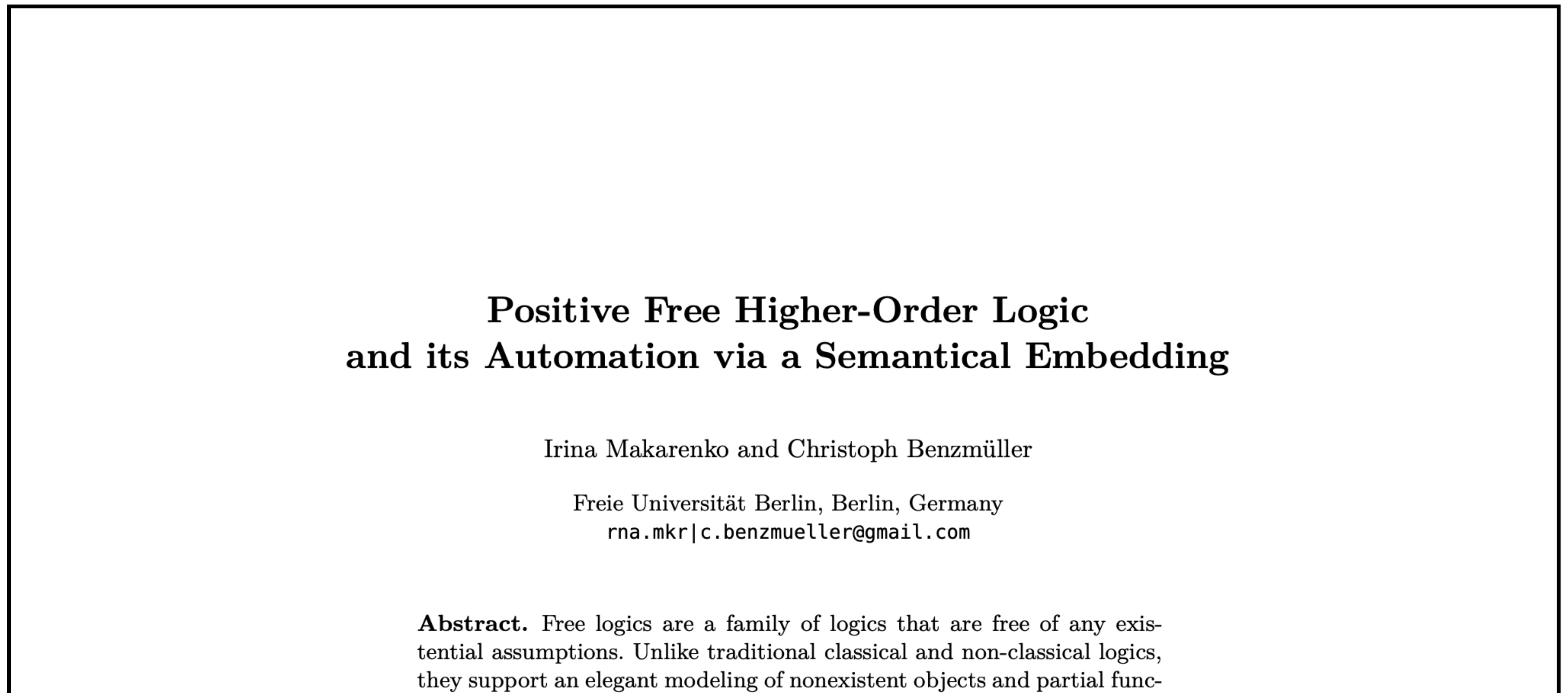
Freie Universität Berlin

Free Higher-Order Logic

and its Automation via a Semantical Embedding



Positive Free Higher-Order Logic and its Automation via a Semantical Embedding

Irina Makarenko and Christoph Benzmüller

Freie Universität Berlin, Berlin, Germany
rna.mkr|c.benzmueller@gmail.com

Abstract. Free logics are a family of logics that are free of any existential assumptions. Unlike traditional classical and non-classical logics, they support an elegant modeling of nonexistent objects and partial func-

Irina Makarenko
EXTENDD
30.04.2025

Structure

- General notion of Free Higher-Order Logic (FHOL)
  - Positive Free Higher-Order Logic (PFHOL)
  - Negative Free Higher-Order Logic (NgFHOL)
  - Neutral Free Higher-Order Logic (NeFHOL)
  - Supervaluational Free Higher-Order Logic (SFHOL)
- Recap: Higher-Order Logic (HOL)
- Free Higher-Order Logic (FHOL)
- Recap: LogiKEy methodology
- Embedding of PFHOL into HOL
- Prior's Paradox
- Summary

General notion of Free Higher-Order Logic (FHOL)

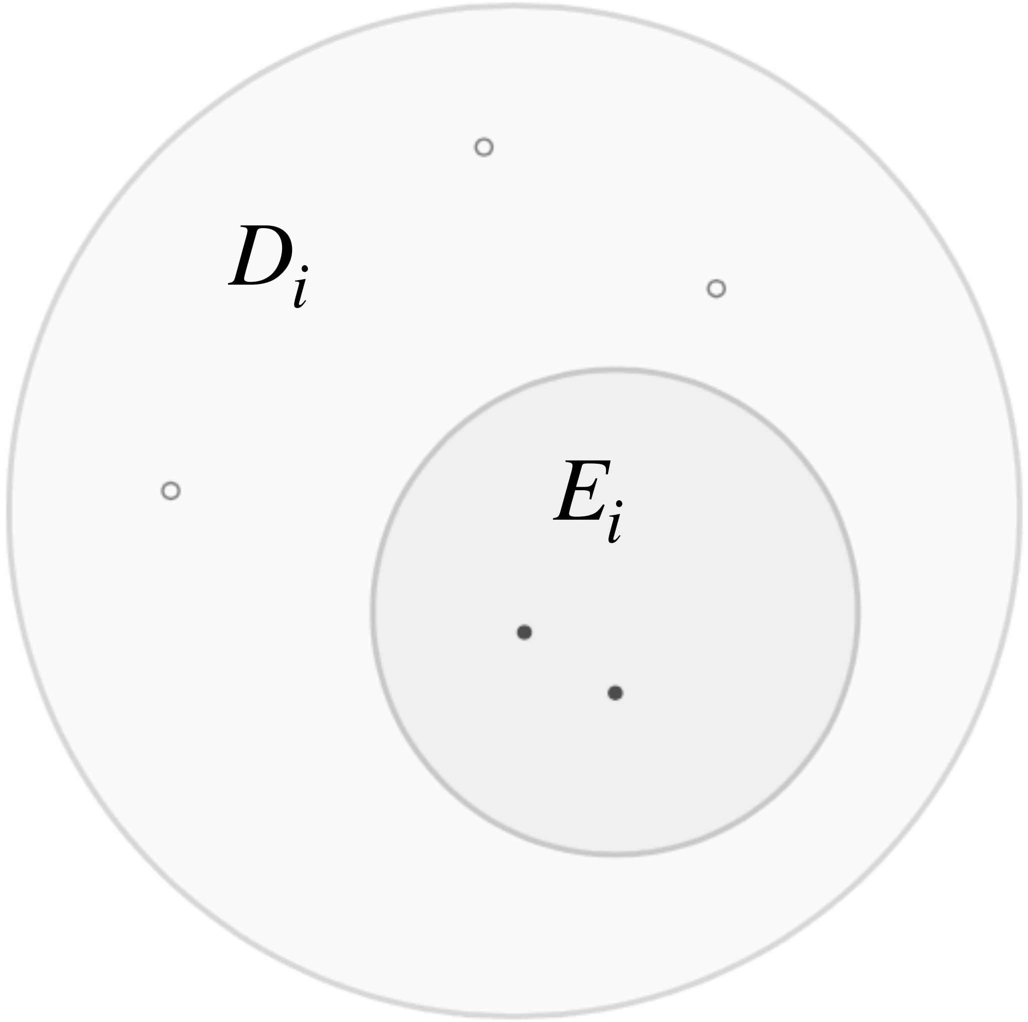
Free logic is a logic that allows for nonexistent objects and partial functions.

Irina Makarenko
EXTENDD
30.04.2025

General notion of Free Higher-Order Logic (FHOL)

Free logic is a logic that allows for nonexistent objects and partial functions.

Quantifiers as well as definite descriptions have existential import.



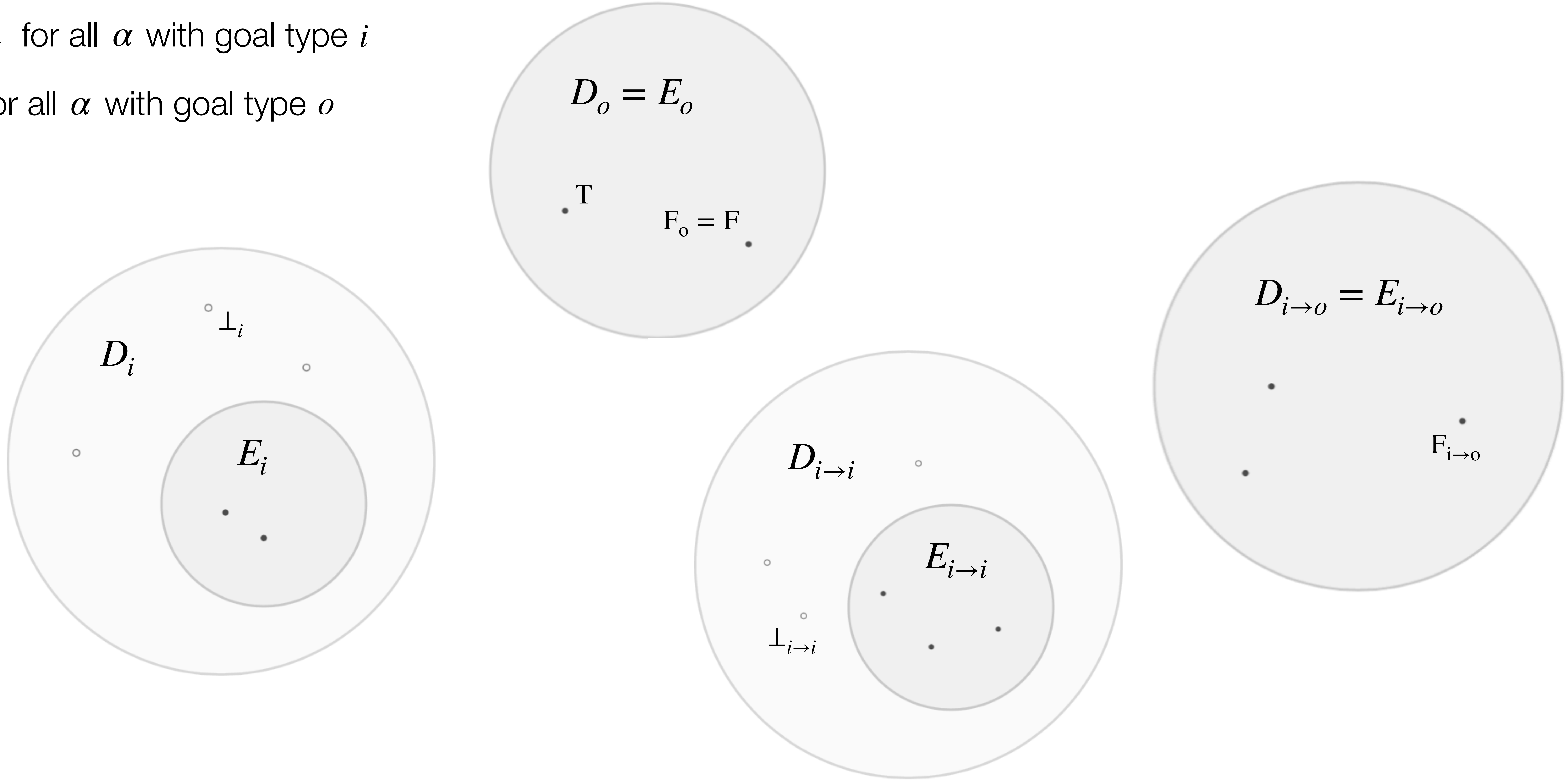
We have one domain for each type which is in the set defined by

$$\alpha, \beta \quad := \quad i \mid o \mid \alpha \to \beta$$

General notion of Free Higher-Order Logic (FHOL)

$\perp_\alpha \in D_\alpha \backslash E_\alpha$ for all α with goal type i

$F_\alpha \in D_\alpha$ for all α with goal type o



Irina Makarenko
EXTENDD
30.04.2025

Positive Free Higher-Order Logic (PFHOL)

A free logic is called positive when atomic formulas with terms that do not refer to anything existing can be evaluated to true but do not have to be.

$$hasLegs(Pegasus) = True$$

$$hasArms(Pegasus) = False$$

Negative Free Higher-Order Logic (NgFHOL)

A free logic is called negative when atomic formulas with terms that do not refer to anything existing are all denied.

$$hasLegs(Pegasus) = False$$

Neutral Free Higher-Order Logic (NeFHOL)

Positive and negative free logics require atomic formulas containing terms that do not refer to anything existing to be either true or false, even in cases where no decision can be made.

$$hasLegs(Pegasus) = \bot$$

$$isSixFeetTall(Pegasus) = \bot$$

$$hasLegs(Pegasus) \land hasLegs(horse) = \bot$$

A free logic is called neutral when atomic formulas with terms that do not refer to anything existing are not assigned any truth value. This inevitably means moving away from a bivalent to a trivalent logic, a logic with three different truth values.

Supervaluational Free Higher-Order Logic (SFHOL)

A supervaluational free logic attempts to account for the trivalent nature of neutral free logic while preserving the principles of classical reasoning.

$$isSixFeetTall(Pegasus) = \bot$$

$$isSixFeetTall(Pegasus) \vee \neg isSixFeetTall(Pegasus) = True$$

Recap: Higher-Order Logic (HOL)

HOL: Simple Syntax

Simple Types: $\alpha, \beta \quad ::= \quad i \mid o \mid (\alpha \to \beta)$

(we may add further base types; types are often not displayed)

Simply Typed λ-Calculus (with constants):

$$\boldsymbol{s, t \quad ::= \quad p_\alpha \quad | \quad X_\alpha \quad | \quad (\lambda X_\alpha s_\beta)_{\alpha\to\beta} \quad | \quad (s_{\alpha\to\beta}\, t_\alpha)_\beta}$$

constants variables lambda abstraction application

abstraction and application interact, e.g.: $((\lambda X\ (p\ X))\ t) \overset{\beta\text{-}reduction}{\longrightarrow} (p\ t)$

HOL defined on Top of Simply Typed λ-Calculus

- add special constant symbols to signature, e.g.

  $\neg_{o\to o} \quad \vee_{o\to o\to o} \quad \Pi_{(\alpha\to o)\to o}$ (or only $=_{\alpha\to\alpha\to o}$)

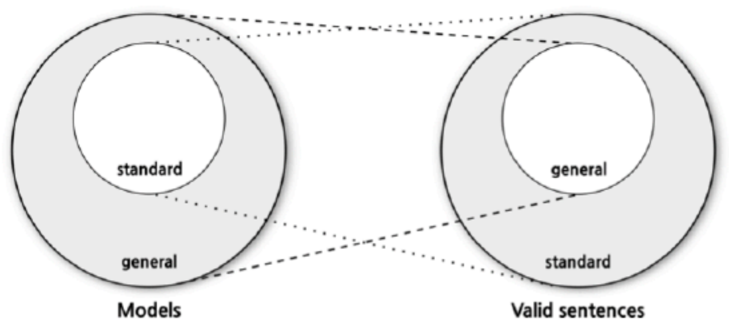
- no binder besides λ needed: $\forall X_\alpha\ s_o$ stands for $\Pi_{(\alpha\to o)\to o}(\lambda X_\alpha\ s_o)$
- $\bot, \top, \to, \leftrightarrow, \exists, =$ can be defined: e.g., $\exists X_\alpha\ s_o$ stands for $\neg\forall X_\alpha\ \neg s_o$

HOL is a language of terms. Terms of type o are called formulas.

HOL: Well Understood Semantics

HOL with standard semantics: incomplete

HOL with Henkin's general semantics: semi-decidable & compact



more model structures . . . fewer valid formulas

Important principles are still valid in Henkin's general models:

- Comprehension (type-restricted): $\forall G\, \exists F\, \forall \overline{X^n}\ F\overline{X^n} = G$
- Boolean Extensionality: $\forall P\, \forall Q\, ((P \leftrightarrow Q) \to P = Q)$
- Functional Extensionality: $\forall F\, \forall G\, ((\forall X\ F\ X = G\ X) \to F = G)$

Note: Any "Henkin-valid" formula is also valid in standard semantics!

Suggested Reading

- Origin [Church, JSL, 1940]
- Henkin's general semantics: [Henkin, JSL, 1950] [Andrews, JSL, 1971, 1972]
- Extensionality&Intensionality: [BenzmüllerEtAl., JSL, 2004] [Muskens, JSL, 2007] [PhD thesis by Steen]

Free Higher-Order Logic (FHOL)

Definition 13. Terms of FHOL are defined based on the following formation rules:

$$
\begin{aligned}
s,t \; ::= \quad & P_\alpha \mid x_\alpha \mid (E!_{\alpha\to o}\, s_\alpha)_o \mid ((=_{\alpha\to\alpha\to o}\, s_\alpha)_{\alpha\to o}\, t_\alpha)_o \mid \\
& (\neg_{o\to o}\, s_o)_o \mid ((\vee_{o\to o\to o} s_o)\, t_o)_o \mid (\forall_{(\alpha\to o)\to o}(\lambda x_\alpha.\, s_o)_{\alpha\to o})_o \mid \\
& (\iota_{(\alpha\to o)\to\alpha}(\lambda x_\alpha.\, s_o)_{\alpha\to o})_\alpha \mid (s_{\alpha\to\beta}\, t_\alpha)_\beta \mid (\lambda x_\alpha.\, s_\beta)_{\alpha\to\beta}
\end{aligned}
$$

with $\alpha, \beta \in \mathcal{T}$.

Definition 14. A *frame* D is a set $\{D_\alpha : \alpha \in \mathcal{T}\}$ of nonempty sets (formally *domains*) D_α such that D_i is chosen freely, $D_o = \{\mathrm{T}, \mathrm{F}\}$ where $\mathrm{T} \neq \mathrm{F}$ and T represents truth and F represents falsehood, and $D_{\alpha\to\beta}$ is the set of all total functions from domain D_α to codomain D_β.

Definition 15. A *subframe* E is a set $\{E_\alpha : \alpha \in \mathcal{T}\}$ of possibly empty sets (formally *domains*) D_α such that $E_\alpha \subset D_\alpha$ for each $\alpha \in \mathcal{T}_i$ and $E_\alpha = D_\alpha$ for each $\alpha \in \mathcal{T}_o$.[19]

Definition 16. A *standard model* is a triple $M = \langle D, E, I \rangle$ where D is a frame, E is a subframe and I is a family of typed interpretation functions, i.e., $I = \{I_\alpha : \alpha \in \mathcal{T}\}$. Each *interpretation function* I_α maps constants of type α to appropriate elements of D_α.

Free Higher-Order Logic (FHOL)

$$
\begin{array}{llll}
I(E!_{\alpha\to o}) & := ex & \in E_{\alpha\to o} & \text{s.t. for all } d \in D_\alpha, \\
 & & & ex(d) = \mathrm{T} \text{ iff } d \in E_\alpha \\
I(=_{\alpha\to\alpha\to o}) & := id & \in E_{\alpha\to\alpha\to o} & \text{s.t. for all } d, d' \in D_\alpha, \\
 & & & id(d, d') = \mathrm{T} \text{ iff } d \text{ is identical to } d' \\
I(\neg_{o\to o}) & := not & \in E_{o\to o} & \text{s.t. } not(\mathrm{T}) = \mathrm{F} \text{ and } not(\mathrm{F}) = \mathrm{T} \\
I(\vee_{o\to o\to o}) & := or & \in E_{o\to o\to o} & \text{s.t. } or(v_1, v_2) = \mathrm{T} \text{ iff } v_1 = \mathrm{T} \text{ or } v_2 = \mathrm{T} \\
I(\forall_{(\alpha\to o)\to o}) & := allq & \in E_{(\alpha\to o)\to o} & \text{s.t. for all } f \in D_{\alpha\to o}, \\
 & & & allq(f) = \mathrm{T} \text{ iff } f(d) = \mathrm{T} \text{ for all } d \in E_\alpha \\
I(\iota_{(\alpha\to o)\to\alpha}) & := desc & \in E_{(\alpha\to o)\to\alpha} & \text{s.t. for all } f \in D_{\alpha\to o}, \\
 & & & desc(f) = d \in E_\alpha \text{ if } f(d) = \mathrm{T} \text{ and} \\
 & & & \text{for all } d' \in E_\alpha\text{: if } f(d') = \mathrm{T}, \text{ then } d' = d, \\
 & & & \text{otherwise } desc(f) = \bot_\alpha \text{ if } \alpha \in \mathcal{T}_i \text{ and} \\
 & & & desc(f) = \mathrm{F}_\alpha \text{ if } \alpha \in \mathcal{T}_o
\end{array}
$$

with $\alpha \in \mathcal{T}$.

Free Higher-Order Logic (FHOL)

The value $[\![s_\alpha]\!]^{M,g}$ of each PFHOL term s_α in a model M under the variable assignment g is an object $d \in D_\alpha$ and is evaluated as follows:

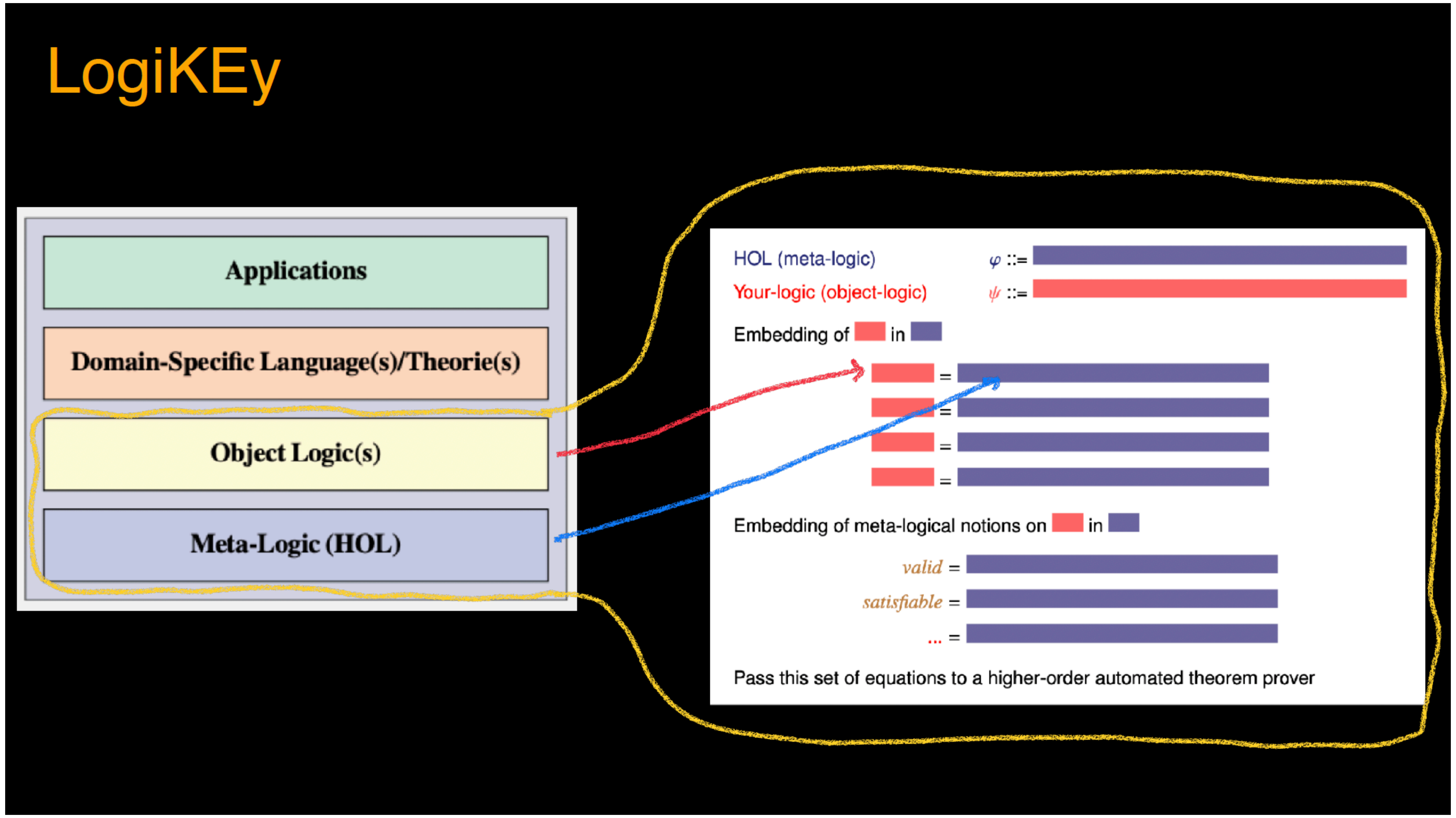
$$
\begin{array}{lll}
[\![P_\alpha]\!]^{M,g} & := & I(P_\alpha) \\
[\![x_\alpha]\!]^{M,g} & := & g(x_\alpha) \\
[\![(s_{\alpha\to\beta}\, t_\alpha)_\beta]\!]^{M,g} & := & [\![s_{\alpha\to\beta}]\!]^{M,g} ([\![t_\alpha]\!]^{M,g}) \\
[\![(\lambda x_\alpha .\, s_\beta)_{\alpha\to\beta}]\!]^{M,g} & := & \text{the function } f \text{ from } D_\alpha \text{ into } D_\beta \\
 & & \text{s.t. for all } d \in D_\alpha\text{: } f(d) = [\![s_\beta]\!]^{M,g[x\to d]}
\end{array}
$$

with $\alpha, \beta \in \mathcal{T}$.

$$
[\![(s_{\alpha \to_i \beta}\, t_\alpha)_\beta]\!]^{M,g} := \begin{cases} [\![s_{\alpha\to_i\beta}]\!]^{M,g} ([\![t_\alpha]\!]^{M,g}) & \text{if } [\![t_\alpha]\!]^{M,g} \in E_\alpha \\ \bot_\beta & \text{else} \end{cases}
$$

with $(\alpha \to_i \beta) \in \mathcal{T}_i$.

Recap: LogiKEy methodology



Irina Makarenko
EXTENDD
30.04.2025

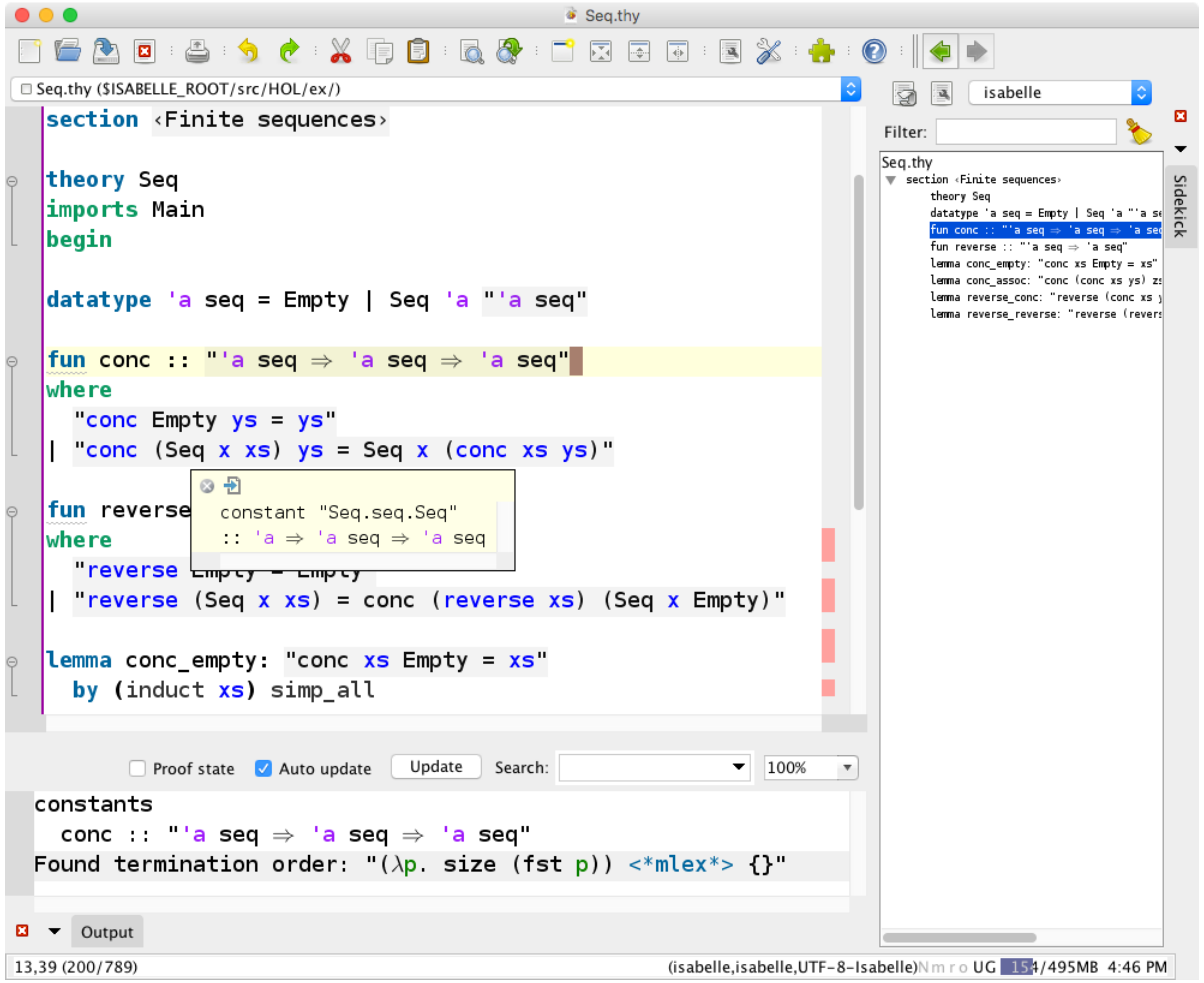
Embedding of PFHOL into HOL

Isabelle/HOL is an interactive proof assistant providing access to the model generator nitpick and the meta-prover sledgehammer that, in turn, invokes third-party resolution provers, SMT solvers, and higher-order provers.

Embedding will be given directly in Isabelle/HOL.

Embedding of PFHOL into HOL

Freie Universität Berlin



source

Irina Makarenko
EXTENDD
30.04.2025

Embedding of PFHOL into HOL


```
typedecl i

consts fExistence :: "'a ⇒ bool" ("E")

consts fUndef :: "'a" ("e")
axiomatization where fUndefIAxiom: "¬E (e::i)"
axiomatization where fFalsehoodBAxiom: "(e::bool) = False"
axiomatization where fTrueAxiom: "E True"
axiomatization where fFalseAxiom: "E False"
```

Embedding of PFHOL into HOL

```
definition fIdentity :: "'a ⇒ 'a ⇒ bool" (infixr "=" 56)
    where "φ = ψ ≡ φ = ψ"

definition fNot :: "bool ⇒ bool" ("¬_" [52]53)
    where "¬φ ≡ ¬φ"
definition fOr :: "bool ⇒ bool ⇒ bool" (infixr "∨" 51)
    where "φ ∨ ψ ≡ φ ∨ ψ"
```

Embedding of PFHOL into HOL

```
definition fForall :: "('a ⇒ bool) ⇒ bool" ("∀")
  where "∀Φ ≡ ∀x. E x ⟶ Φ x"
definition fForallBinder:: "('a ⇒ bool) ⇒ bool" (binder "∀" [8]9)
  where "∀x. φ x ≡ ∀φ"

definition fThat :: "('a ⇒ bool) ⇒ 'a" ("I")
  where "IΦ ≡ if ∃x. E x ∧ Φ x ∧ (∀y. (E y ∧ Φ y) ⟶ (y = x))
             then THE x. E x ∧ Φ x
             else e"
definition fThatBinder:: "('a ⇒ bool) ⇒ 'a" (binder "I" [8]9)
  where "Ix. φ x ≡ Iφ"
```

Embedding of PFHOL into HOL

```
definition fAnd :: "bool ⇒ bool ⇒ bool" (infixr "∧" 52)
  where "φ ∧ ψ ≡ ¬(¬φ ∨ ¬ψ)"
definition fImp :: "bool ⇒ bool ⇒ bool" (infixr "→" 49)
  where "φ → ψ ≡ ¬φ ∨ ψ"
definition fEquiv :: "bool ⇒ bool ⇒ bool" (infixr "↔" 50)
  where "φ ↔ ψ ≡ φ → ψ ∧ ψ → φ"
definition fExists :: "('a ⇒ bool) ⇒ bool" ("∃")
  where "∃Φ ≡ ¬(∀(λy. ¬(Φ y)))"
definition fExistsBinder :: "('a ⇒ bool) ⇒ bool" (binder "∃" [8]9)
  where "∃x. φ x ≡ ∃φ"
```

Prior's Paradox

$$Q\forall p(Qp \to \neg p) \;\to\; \exists p(Qp \land p) \;\land\; \exists p(Qp \land \neg p)$$

Reading Qp as, for instance, 'Kaplan says at midnight that p', Prior's paradox says that if Kaplan says at midnight that everything he says at midnight is false, then he says something true at midnight and also something false at midnight.

```
lemma "(Q (∀p. (Q p ⟶ (¬p)))) ⟶ ((∃p. Q p ∧ p) ∧ (∃p. Q p ∧ (¬p)))"
```

Prior’s Paradox

```
axiomatization where fTrueAxiom: "E True"
axiomatization where fFalseAxiom: "E False"

lemma "(Q (∀p. (Q p ⟶ (¬p)))) ⟶ ((∃p. Q p ∧ p) ∧ (∃p. Q p ∧ (¬p)))"
  using Defs by (smt fFalseAxiom fTrueAxiom)
```

Prior's Paradox

```
lemma "(Q (∀p. (Q p ⟶ (¬p)))) ⟶ ((∃p. Q p ∧ p) ∧ (∃p. Q p ∧ (¬p)))"
  nitpick [user_axioms=true, show_all, format=2]
  oops
```

```
Nitpick found a counterexample for card i = 3:

  Free variable:
    Q = (λx. _)(True := True, False := True)
  Constants:
    E = (λx. _)(True := True, False := False)
    E = (λx. _)(i₁ := False, i₂ := False, i₃ := True)
    e = i₂
    e = False
```

Summary

We exploited a shallow semantical embedding (SSE) for reducing the automation of free higher-order logic to reasoning within a classical higher-order logic framework.

SSEs allow us to profit from the strength of long-standing reasoning systems for establishing the correctness of theories in non-classical logics like free logic.

Free logic is well suited to represent abstract objects and to support hypothetical reasoning with fictive (and concrete) entities, and can therefore be applied in metaphysics, ethics, and law.

Thank you

Literature:

Irina Makarenko. Free Higher-Order Logic – Notion, Definition and Embedding in HOL. Master's thesis, Freie Universität Berlin, 2020.

Christoph Benzmüller and Dana Scott. Automating Free Logic in HOL, with an Experimental Application in Category Theory. *Journal of Automated Reasoning*, pages 1–20, 2019.

Christoph Benzmüller, Xavier Parent, and Leendert van der Torre. Designing Normative Theories for Ethical and Legal Reasoning: LogiKEy Framework, Methodology, and Tool Support. Artificial Intelligence, 287:103348, 2020.

William M. Farmer. A Partial Functions Version of Church's Simple Theory of Types. *Journal of Symbolic Logic*, 55:1269–1291, 1990.

Ermanno Bencivenga. *Free Semantics*, volume 47 of *Boston Studies in the Philosophy of Science*, pages 31–48. Springer Netherlands, Dordrecht, 1981.