Utilizing proof assistant systems and the LogiKEy methodology for experiments on free logic in HOL

Christoph Benzmüller
U Bamberg & FU Berlin

jww colleagues, students & in particular: Dana Scott

ExtenDD Seminar, January 8, 2025

Initial Comment

Free Logic

Is interesting and relevant, sure, but there have been relatively few attempts so far at implementation & automation, and few concrete demonstrations of it's practical use; the danger of this is that it may be considered as only theoretically interesting.

In this talk I demonstrate/report

- How free logic(s) can be easily implemented and automated
- How free logic(s) can used in concrete case studies to reveal interesting aspects

The key to all this is the **LogiKEy methodology** for logic-pluralistic knowledge representation and reasoning.

Talk Structure

Methodology

- Universal (Meta-)Logical Reasoning & Computational Metaphysics
- Logic-Pluralistic KR&R: LogiKEy

Free first-order logic in HOL

- Implementation in HOL using LogiKEy
- Application in Category Theory (with Dana Scott)

Free higher-order logic in HOL

Conclusion

Methodological development

Early attempts at
Universal (Meta-)Logical Reasoning

Studies in
Computational Metaphysics

Universal (Meta-)Logical Reasoning

"If we had it [a *characteristica universalis*], we should be able to reason in metaphysics and morals in much the same way as in geometry and analysis."

(Leibniz, 1677)

Springer Link

Logica Universalis

Published: 27 May 2012

Quantified Multimodal Logics in Simple Type Theory

Christoph Benzmüller & Lawrence C. Paulson

Logica Universalis 7, 7–20 (2013) | Cite this article

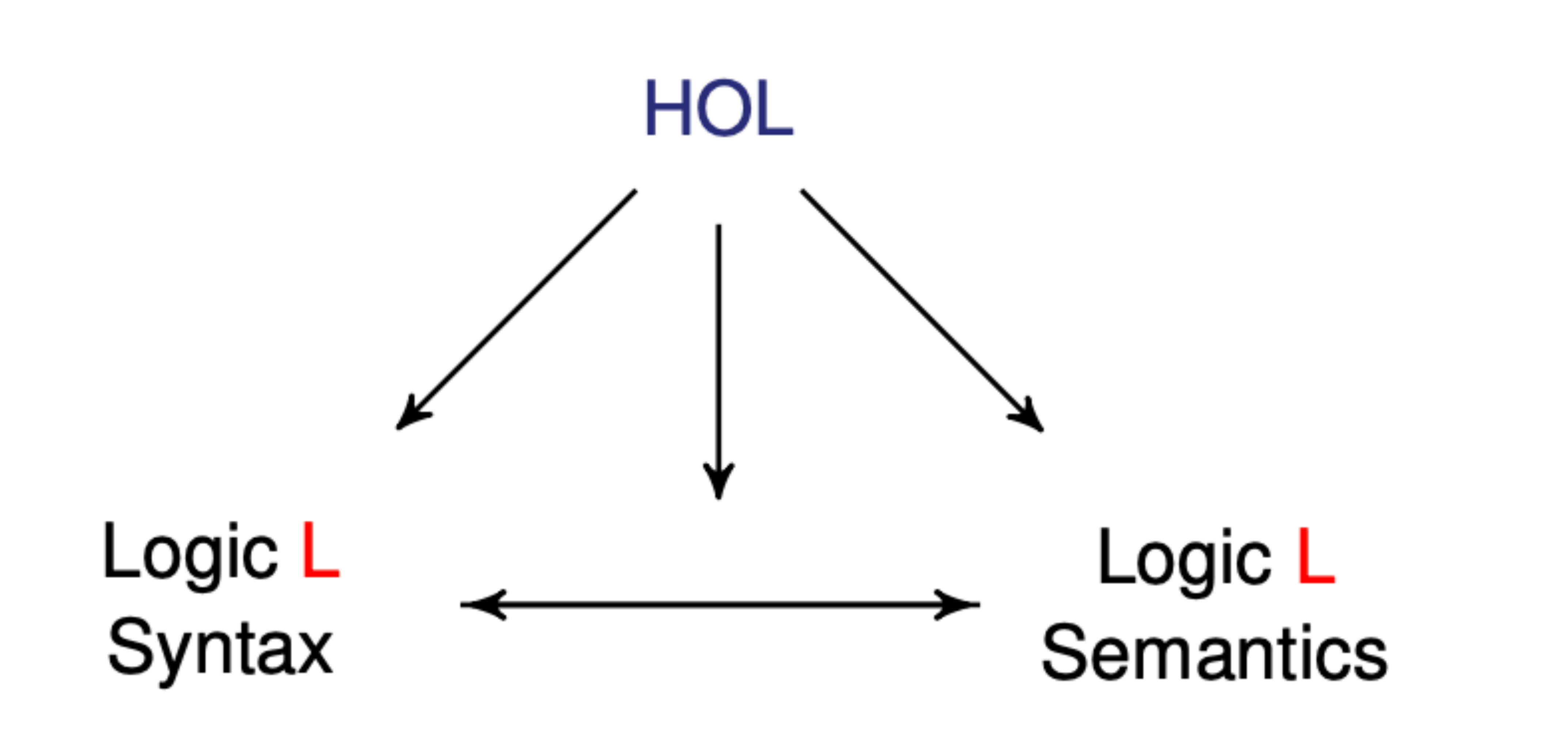
ELSEVIER

Science of Computer Programming

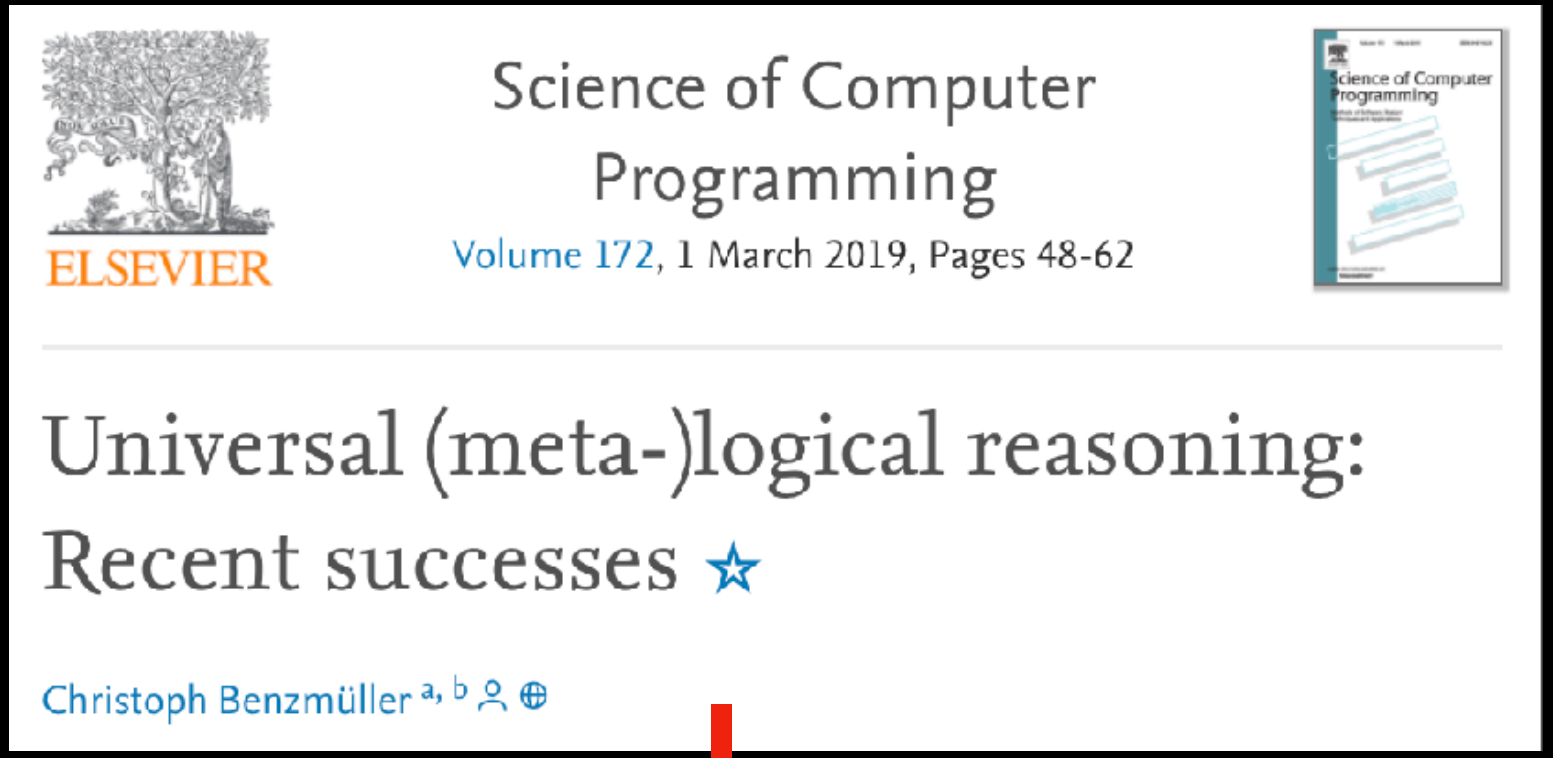
Volume 172, 1 March 2019, Pages 48-62

Universal (meta-)logical reasoning: Recent successes ☆

Christoph Benzmüller [a, b]



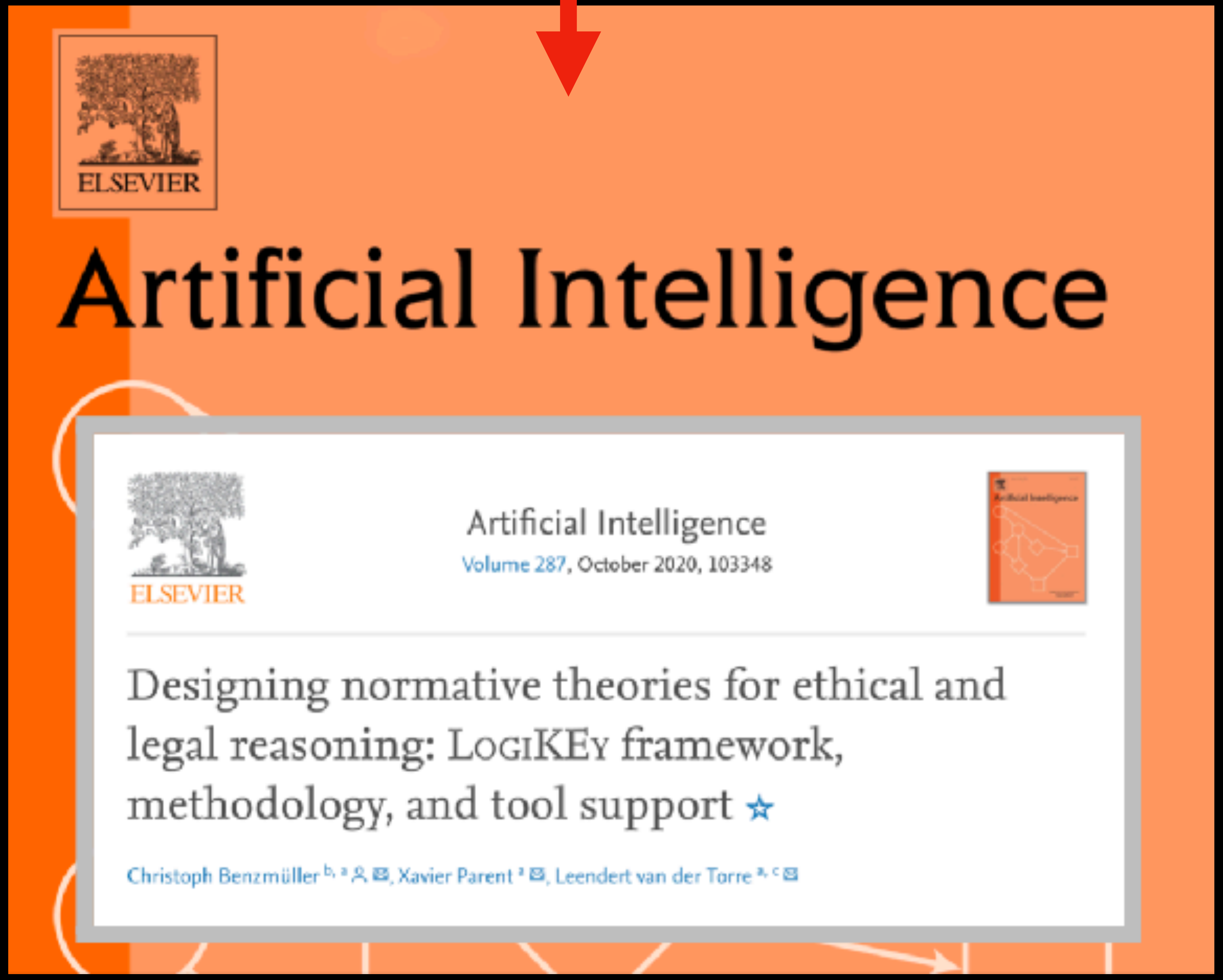
Logic-Pluralistic KR&R Methodology: LogiKEy


ELSEVIER

Science of Computer Programming
Volume 172, 1 March 2019, Pages 48-62

Universal (meta-)logical reasoning: Recent successes ☆

Christoph Benzmüller [a, b]


ELSEVIER

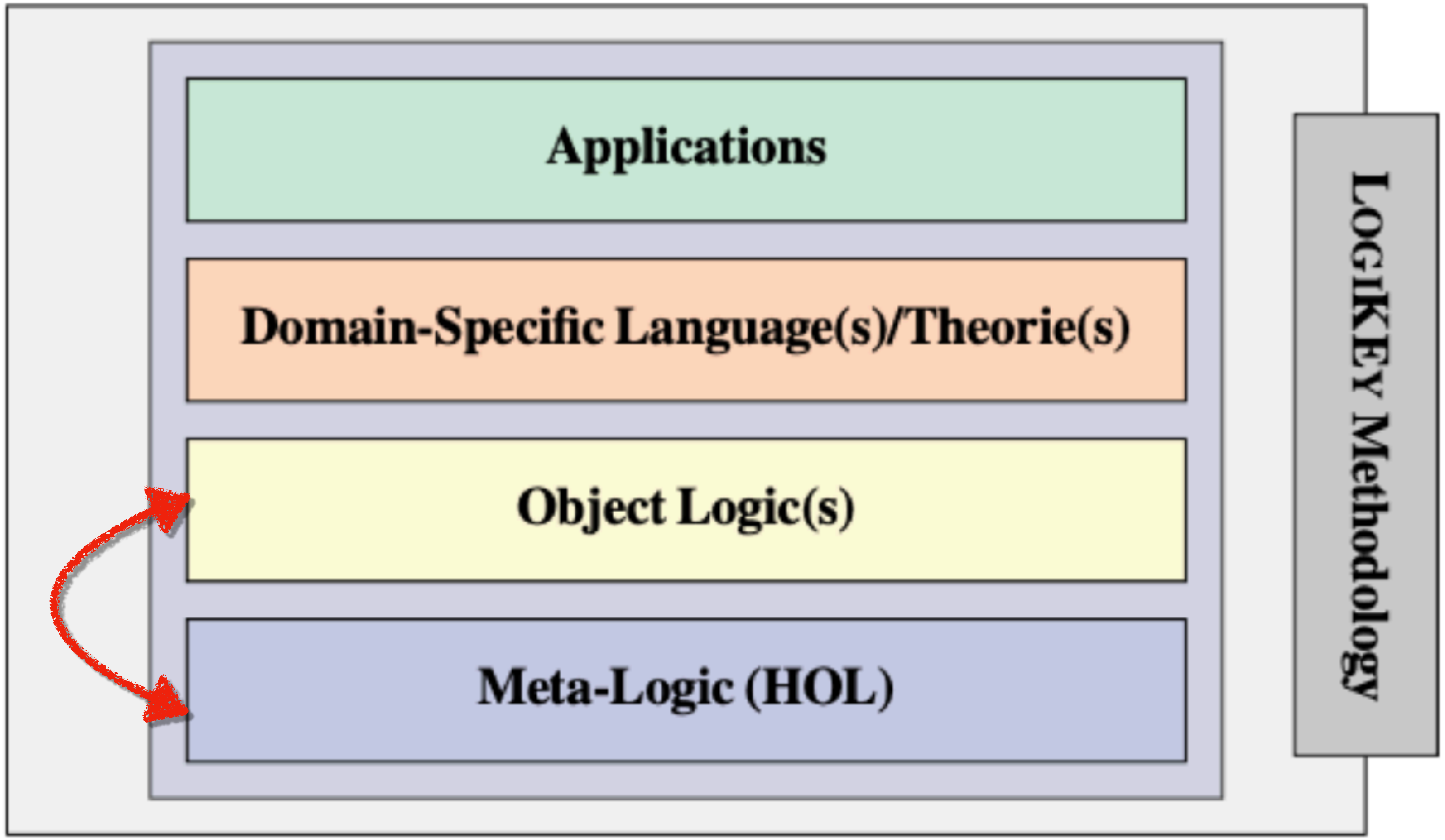
Artificial Intelligence

ELSEVIER

Artificial Intelligence
Volume 287, October 2020, 103348

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]



Different Layers — Flexibility (except HOL) — Parallel & Cyclic Development

Logic-Pluralistic KR&R Methodology: LogiKEy

Metaphysics

Holy Logic: Computer Scientists 'Prove' God Exists

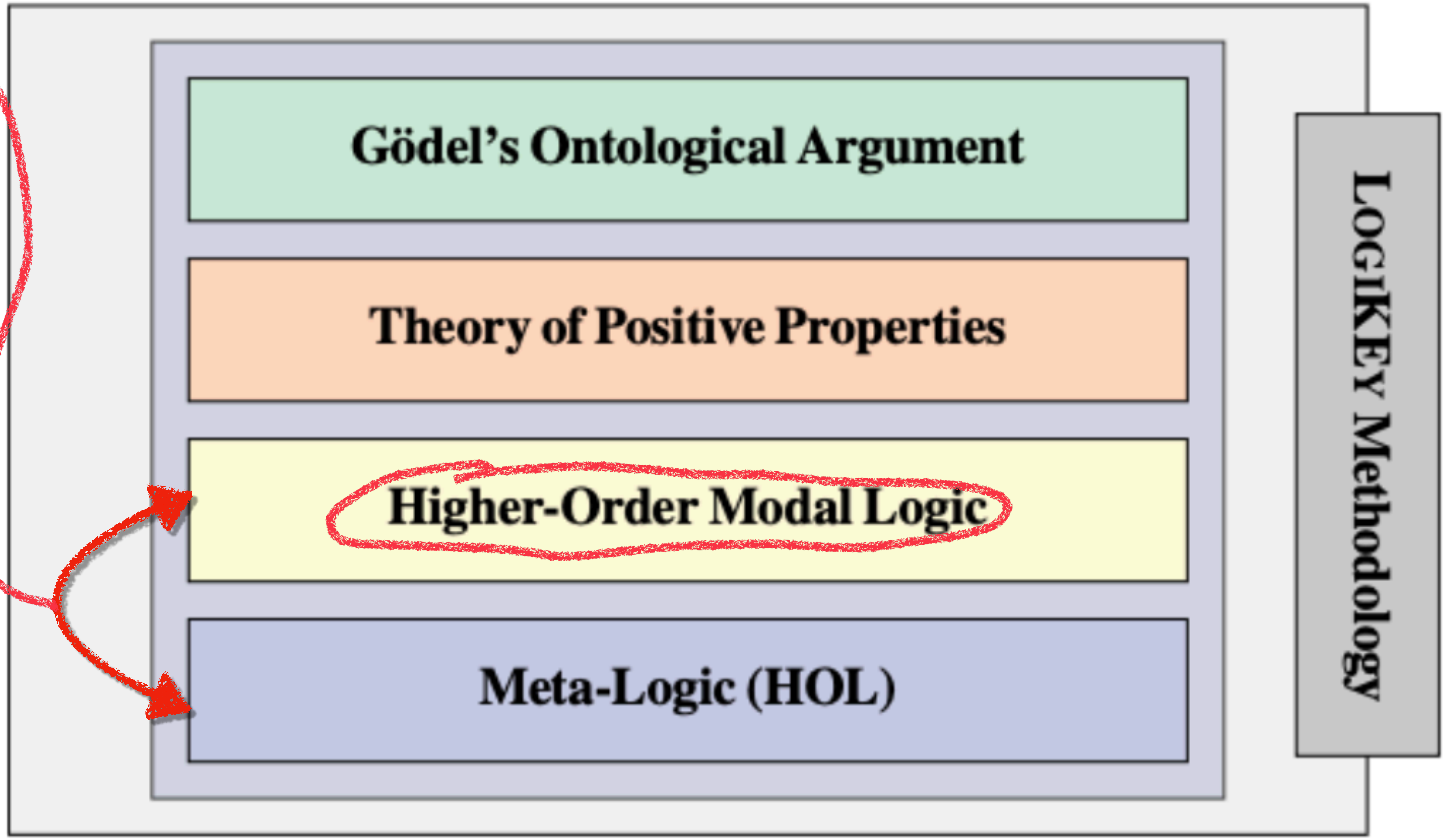
By David Knight

Austrian mathematician Kurt Gödel kept his proof of God's existence a secret for decades. Now two scientists say they have proven it mathematically using a computer.

Two scientists have formalized a theorem regarding the existence of God penned by mathematician Kurt Gödel. But the God angle is somewhat of a red herring -- the real step forward is the example it sets of how computers can make scientific progress simpler.

Automating Gödel's Ontological Proof of God's Existence with Higher-order Automated Theorem Provers

Authors	Christoph Benzmüller, Bruno Woltzenlogel Paleo
Pages	93 - 98
DOI	10.3233/978-1-61499-419-0-93
Series	Frontiers in Artificial Intelligence and Applications
Ebook	Volume 263: ECAI 2014



Different Layers — Flexibility (except HOL) — Parallel & Cyclic Development

Logic-Pluralistic KR&R Methodology: LogiKEy

Metaphysics


Home Video Themen | Forum English DER SPIEGEL | SPIEGEL TV | Abo | Shop RSS Mobile Newsletter

SPIEGEL ONLINE INTERNATIONAL

Front Page World Europe Germany Business Zeitgeist Newsletter

Holy Logic: Computer Scientists 'Prove' God Exists

By David Knight

Austrian mathematician Kurt Gödel kept his proof of God's existence a secret for decades. Now two scientists say they have proven it mathematically using a computer.

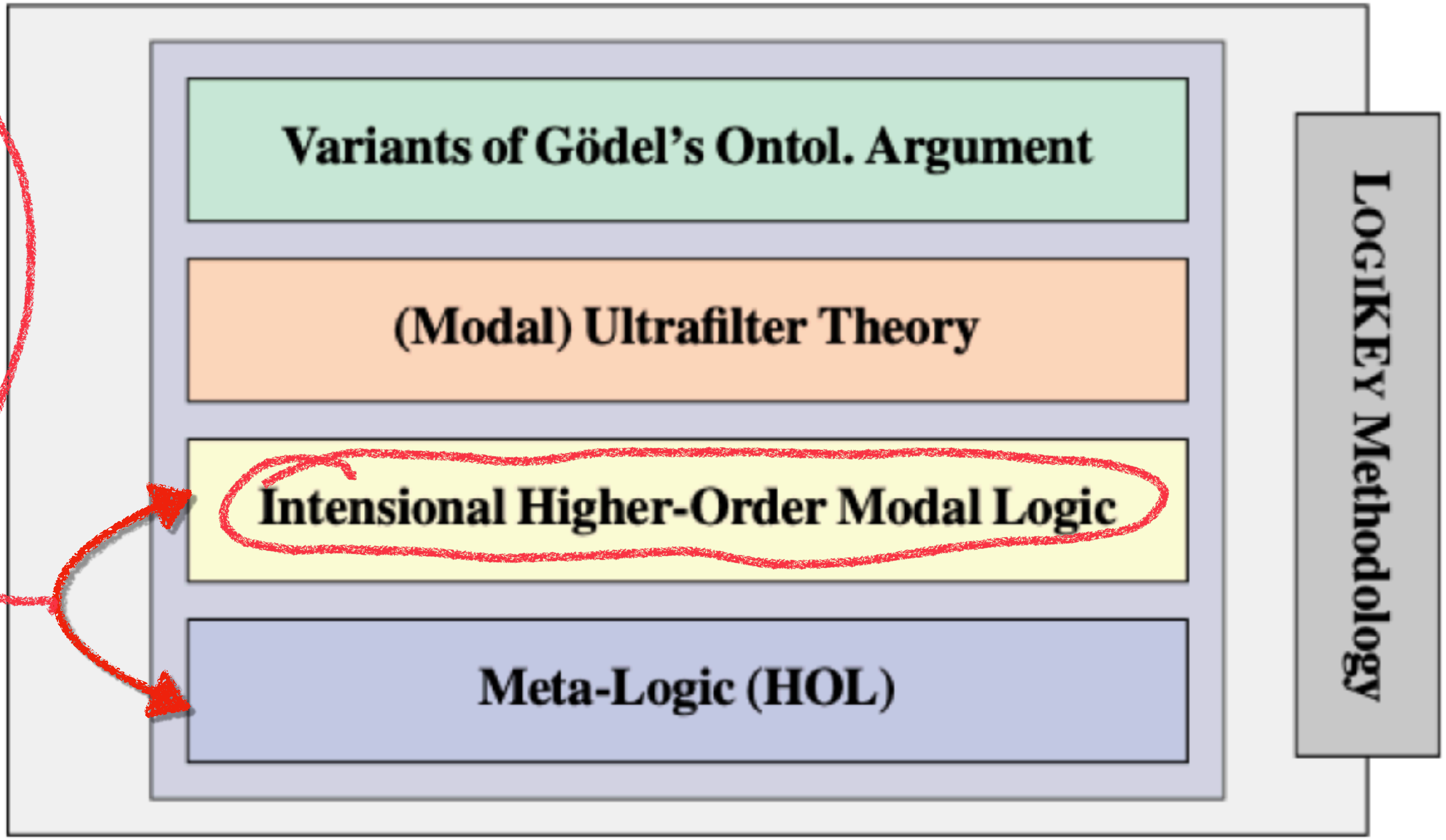
Two scientists have formalized a theorem regarding the existence of God penned by mathematician Kurt Gödel. But the God angle is somewhat of a red herring -- the real step forward is the example it sets of how computers can make scientific progress simpler.

Bulletin of the Section of Logic
Volume 49/2 (2020), pp. 127–148
http://dx.doi.org/10.18778/0138-0680.2020.08

C O P E
Member since 2018
JM13714

Christoph Benzmüller, David Fuenmayor

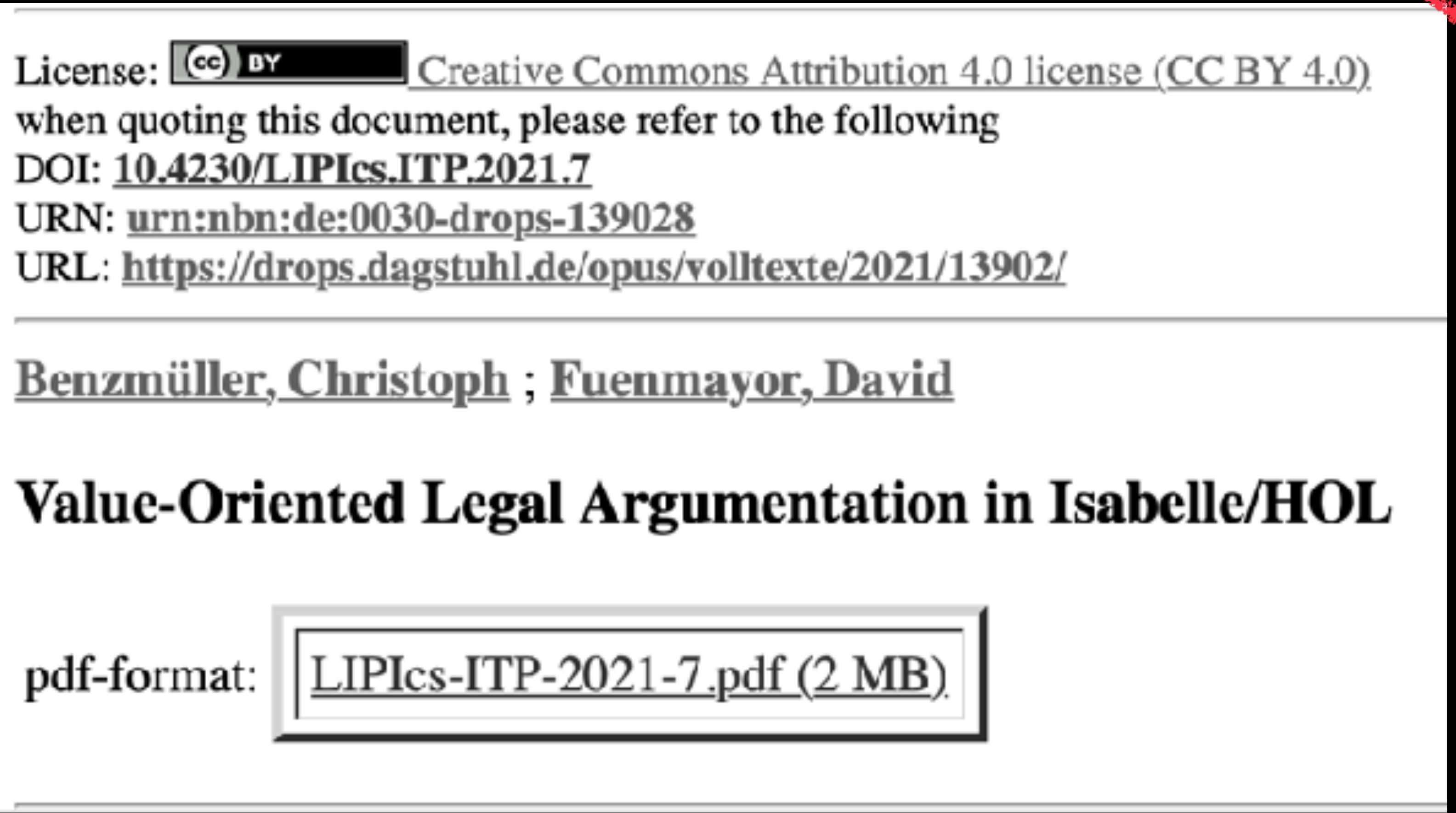
COMPUTER-SUPPORTED ANALYSIS OF POSITIVE PROPERTIES, ULTRAFILTERS AND MODAL COLLAPSE IN VARIANTS OF GÖDEL's ONTOLOGICAL ARGUMENT



Different Layers — Flexibility (except HOL) — Parallel & Cyclic Development

Logic-Pluralistic KR&R Methodology: LogiKEy

Law & Ethics



License: Creative Commons Attribution 4.0 license (CC BY 4.0)
when quoting this document, please refer to the following
DOI: 10.4230/LIPIcs.ITP.2021.7
URN: urn:nbn:de:0030-drops-139028
URL: https://drops.dagstuhl.de/opus/volltexte/2021/13902/

Benzmüller, Christoph ; Fuenmayor, David

Value-Oriented Legal Argumentation in Isabelle/HOL

pdf-format: LIPIcs-ITP-2021-7.pdf (2 MB)

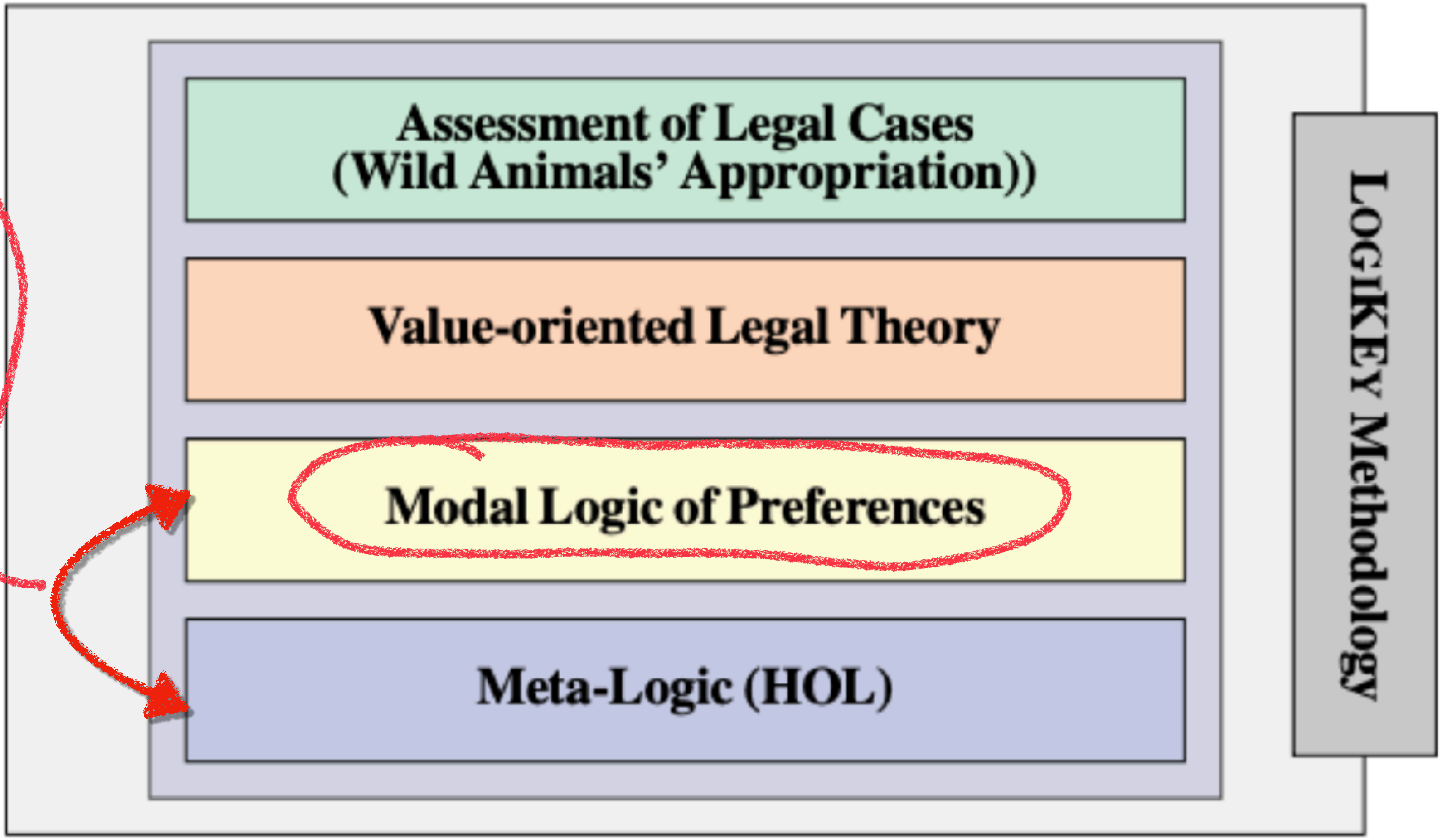
Open Access Article

Modelling Value-Oriented Legal Reasoning in LogiKEy

by Christoph Benzmüller [1,2,*], David Fuenmayor [1,2] and Bertram Lomfeld [3]

[1] AI Systems Engineering, University of Bamberg, 96045 Bamberg, Germany
[2] Department of Mathematics and Computer Science, Freie Universität Berlin, 14195 Berlin, Germany
[3] Department of Law, Freie Universität Berlin, 14195 Berlin, Germany
[*] Author to whom correspondence should be addressed.

Logics **2024**, *2*(1), 31-78; https://doi.org/10.3390/logics2010003



Different Layers — Flexibility (except HOL) — Parallel & Cyclic Development

Logic-Pluralistic KR&R Methodology: LogiKEy

Category Theory

International Congress on Mathematical Software
↳ ICMS 2016: **Mathematical Software – ICMS 2016** pp 43–50 | Cite as

Automating Free Logic in Isabelle/HOL

Christoph Benzmüller & Dana Scott

Springer Link

Published: 01 January 2019

Automating Free Logic in HOL, with an Experimental Application in Category Theory

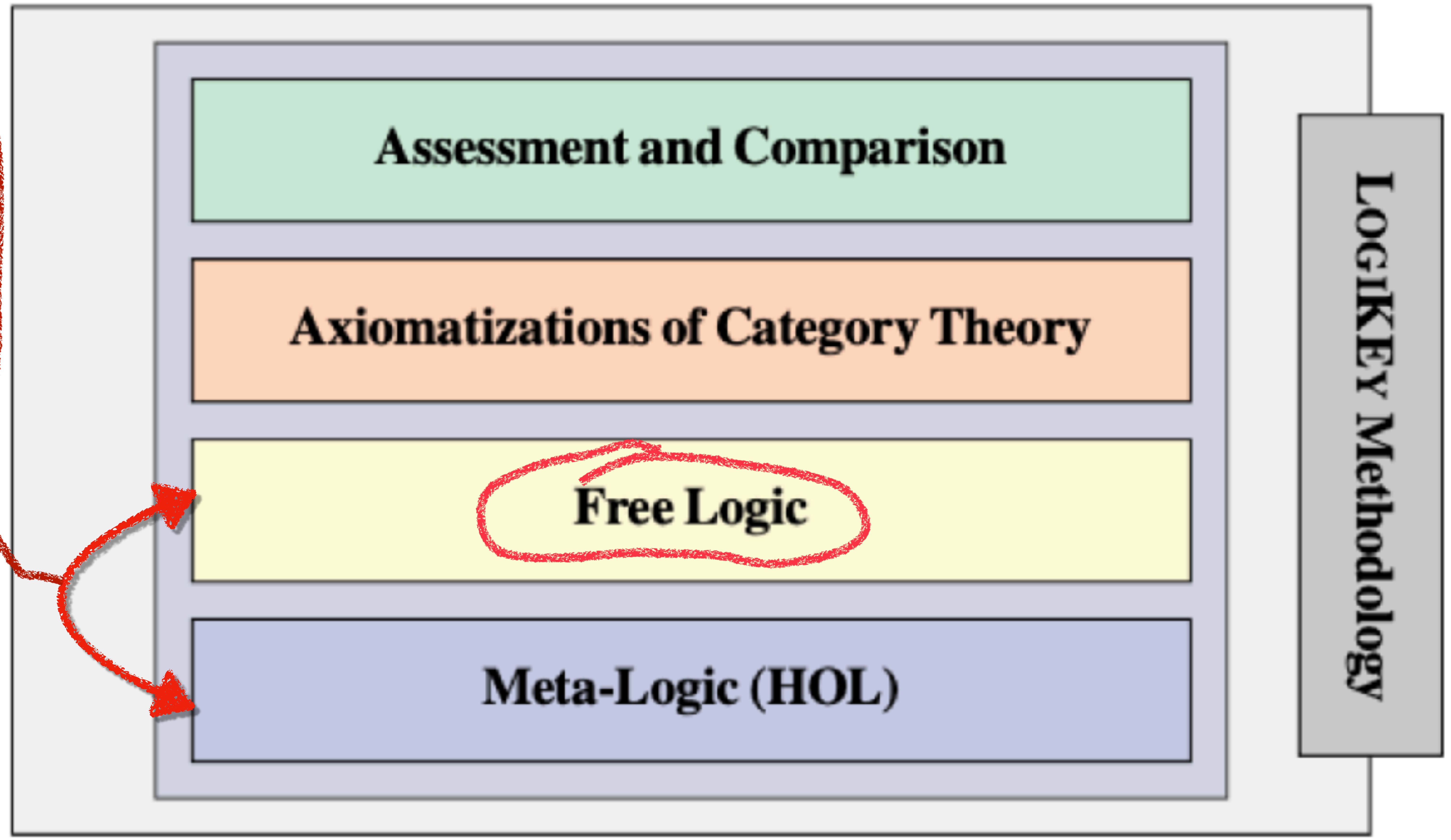
Christoph Benzmüller & Dana S. Scott

Journal of Automated Reasoning **64**, 53–72 (2020) | Cite this article

International Conference on Relational and Algebraic Methods in Computer Science
↳ RAMiCS 2020: **Relational and Algebraic Methods in Computer Science** pp 302–317

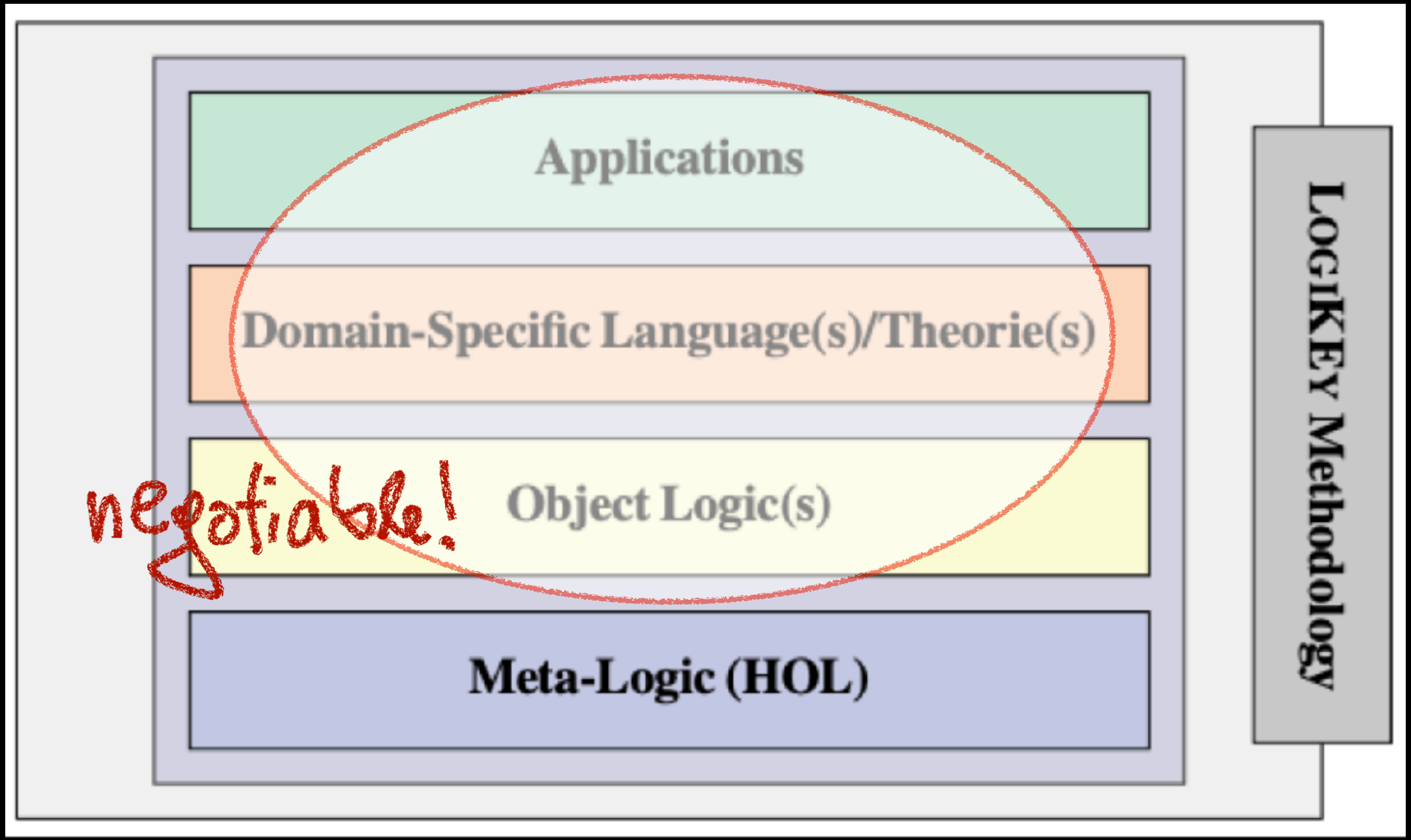
Computer-Supported Exploration of a Categorical Axiomatization of Modeloids

Lucca Tiemens, Dana S. Scott, Christoph Benzmüller & Miroslav Benda



Different Layers — Flexibility (except HOL) — Parallel & Cyclic Development

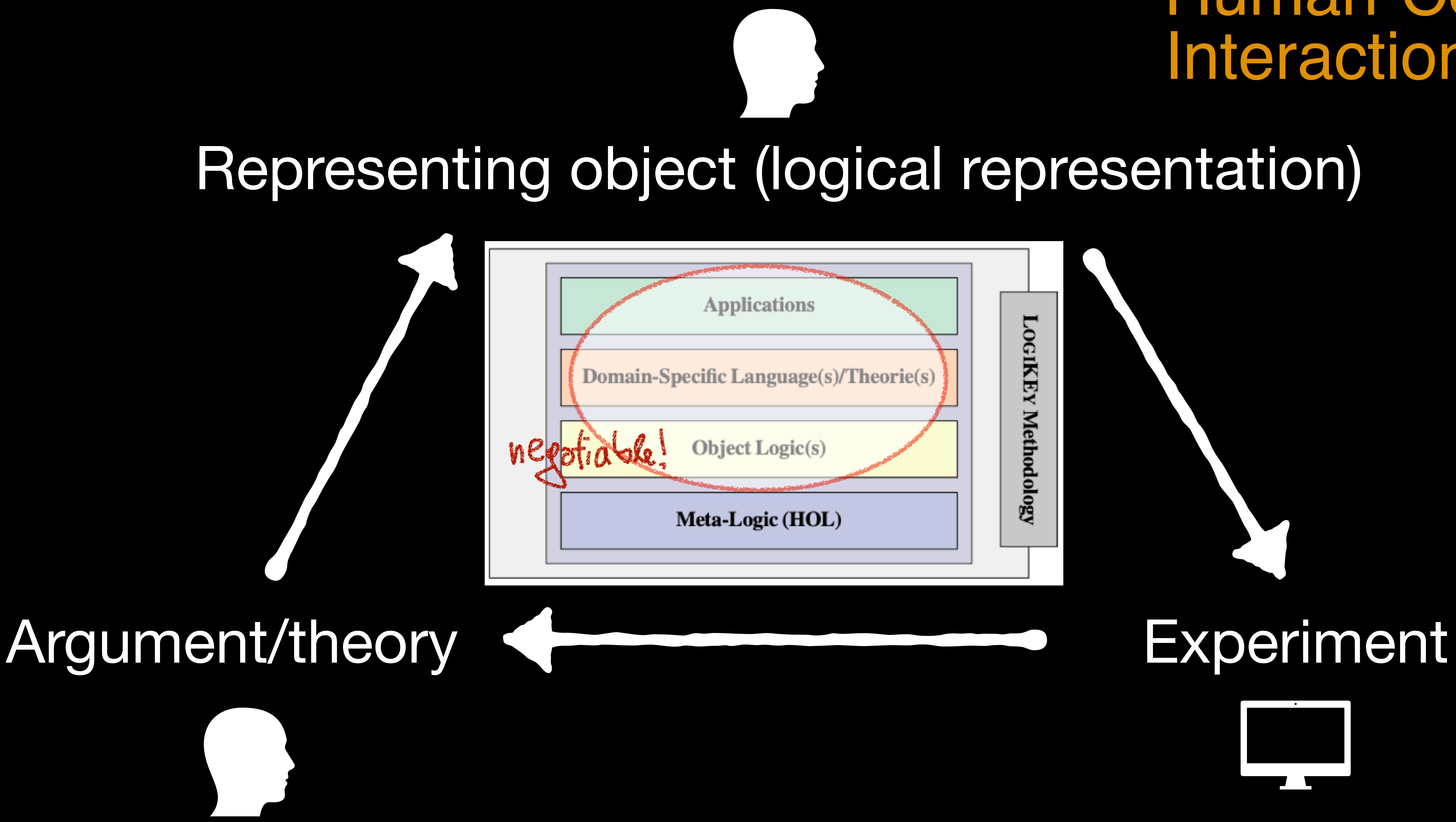
Logic-Pluralistic KR&R Methodology: LogiKEy



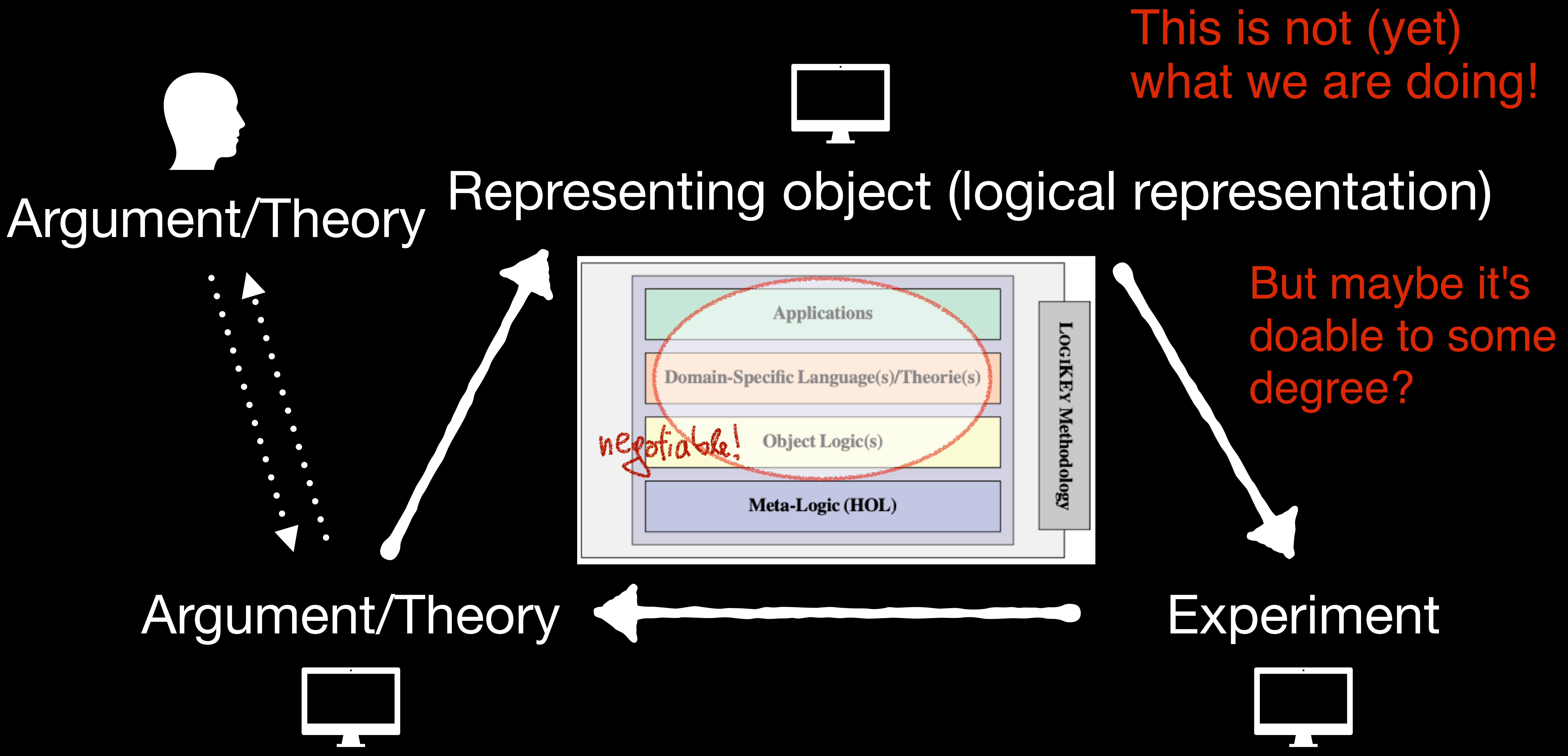
LogiKEy enables both: applications and meta-logical studies.
Highly useful and relevant for the exploration of logics.

Logic-Pluralistic KR&R Methodology: LogiKEy

Human-Computer Interaction



Logic-Pluralistic KR&R Methodology: LogiKEy



Experiments in Computational Metaphysics:

ECAI 2013, IJCAI & KI 2016, KI 2017
(with Bruno Woltzenlogel-Paleo)

Bulletin of the Section of Logic
Volume 49/2 (2020), pp. 127–148
http://dx.doi.org/10.18778/0138-0680.2020.08

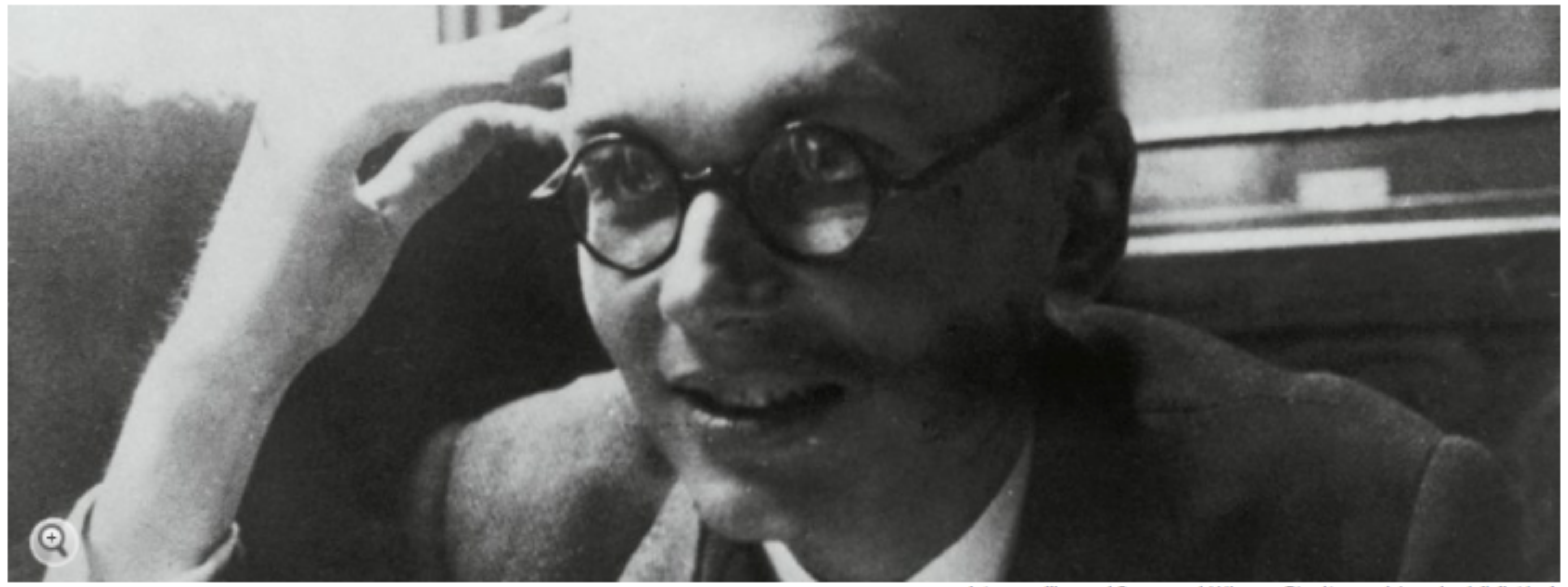
Christoph Benzmüller, David Fuenmayor

COMPUTER-SUPPORTED ANALYSIS OF POSITIVE PROPERTIES, ULTRAFILTERS AND MODAL COLLAPSE IN VARIANTS OF GÖDEL's ONTOLOGICAL ARGUMENT

. . .

Holy Logic: Computer Scientists 'Prove' God Exists

By David Knight



picture-alliance/ Imagno/ Wiener Stadt- und Landesbibliothek

Austrian mathematician Kurt Gödel kept his proof of God's existence a secret for decades. Now two scientists say they have proven it mathematically using a computer.

Two scientists have formalized a theorem regarding the existence of God penned by mathematician Kurt Gödel. But the God angle is somewhat of a red herring -- the real step forward is the example it sets of how computers can make scientific progress simpler.

[Submitted on 13 Feb 2022]

A Simplified Variant of Gödel's Ontological Argument

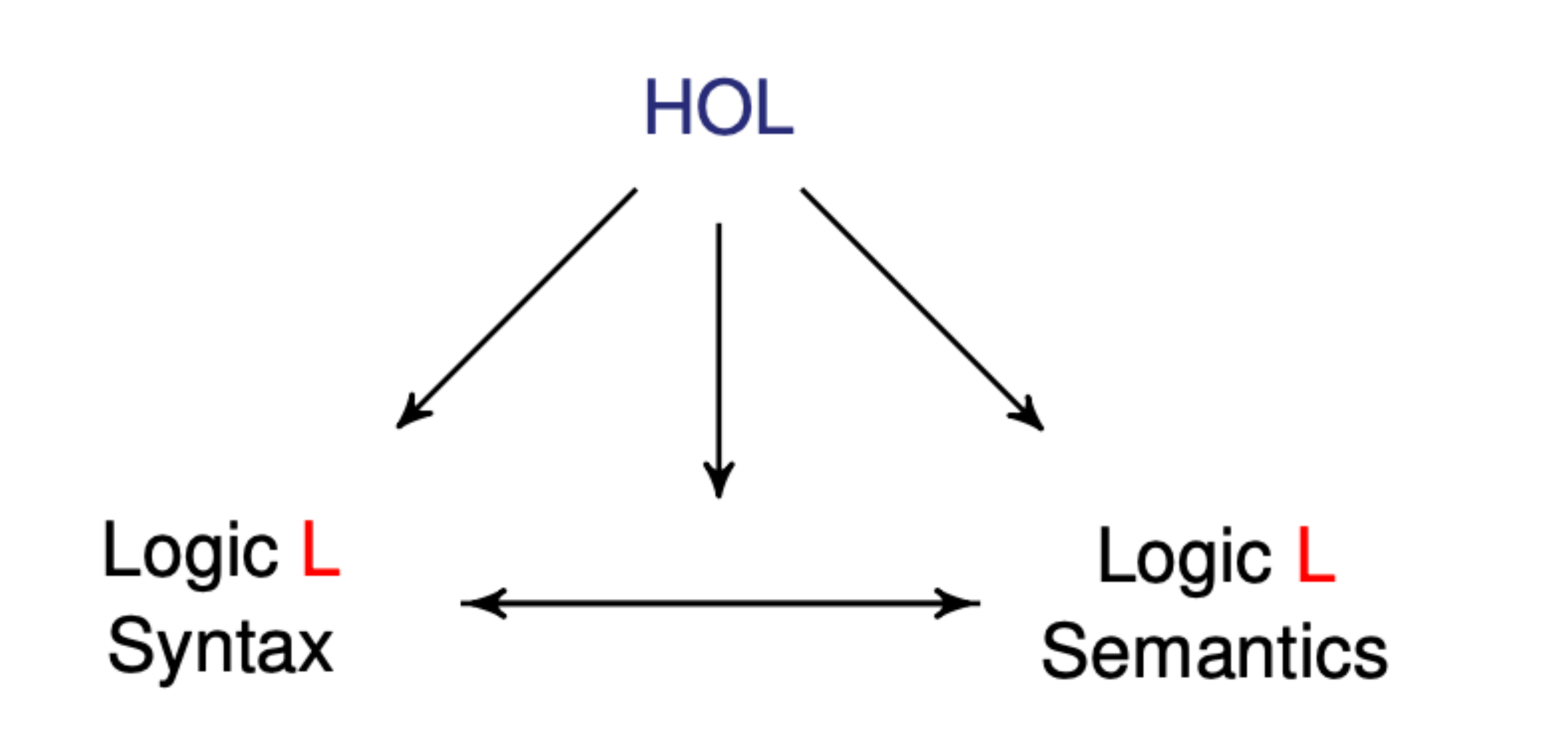
Christoph Benzmüller

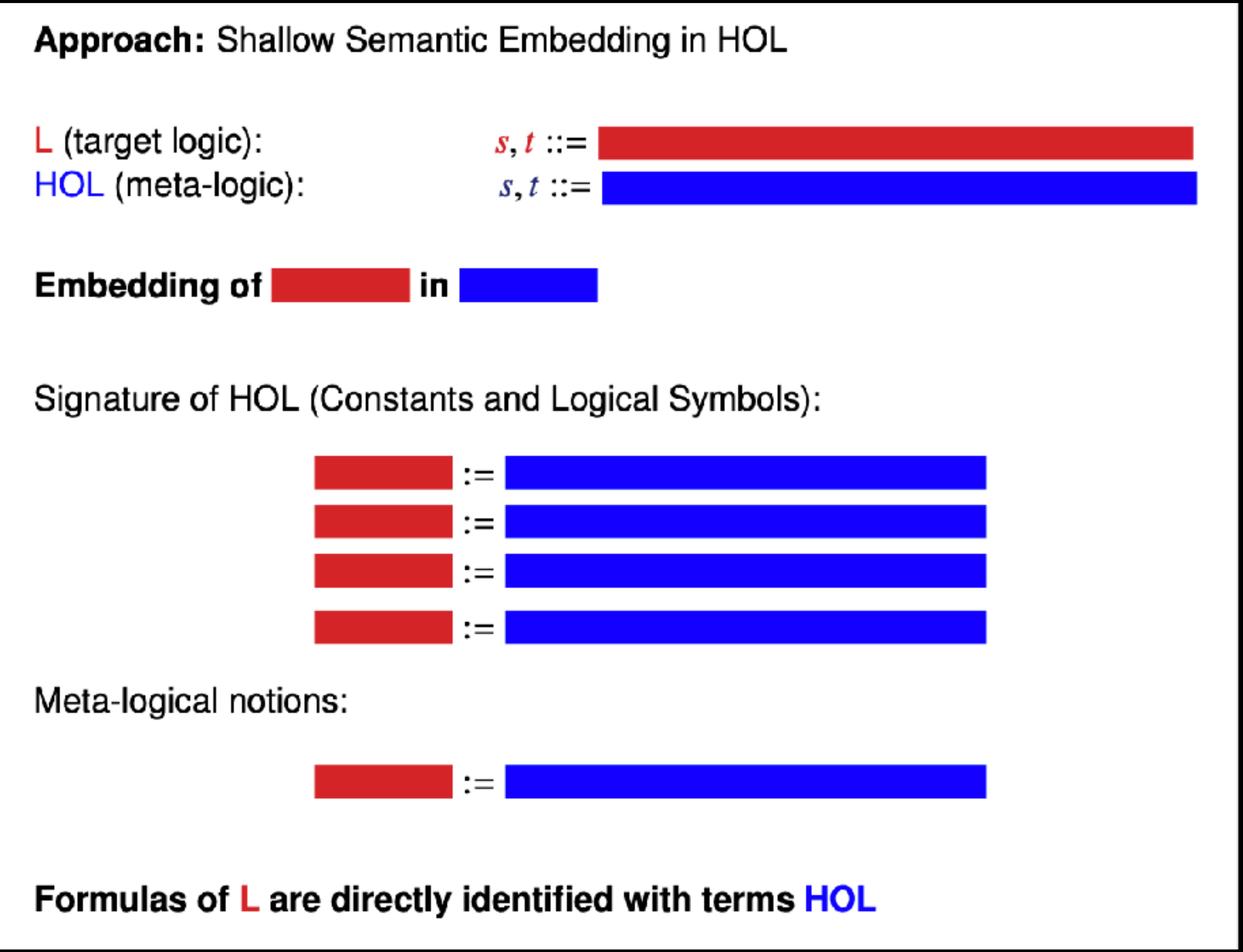
Universal (Meta-)Logical Reasoning in HOL

via Shallow Embeddings in Higher-Order Logic (HOL)

"If we had it [a *characteristica universalis*], we should be able to reason in metaphysics and morals in much the same way as in geometry and analysis."

(Leibniz, 1677)





defining equations are passed to HOL theorem prover(s)

Classical Higher-Order Logic (HOL)

Expressivity	FOL	HOL	Example
Quantification over			
- Individuals	✓	✓	$\forall X\ p(f(X))$
- Functions	-	✓	$\forall F\ p(F(a))$
- Predicates/Sets/Rels	-	✓	$\forall P\ P(f(a))$
Unnamed			
- Functions	-	✓	$\lambda X\ X$
- Predicates	-	✓	$\lambda X\ (X = X)$
Statements about			
- Funcs/Preds/Sets/Rels	-	✓	$reflexive \leftrightarrow$
Powerful definitions	-	✓	$reflexive := \lambda R\ \forall X\ (R\ X\ X)$

Classical Higher-Order Logic (HOL)

Expressivity	FOL	HOL	Example
Quantification over			
- Individuals	✓	✓	$\forall X\ p(f(X))$
- Functions	-	✓	$\forall F\ p(F(a))$
- Predicates/Sets/Rels	-	✓	$\forall P\ P(f(a))$
Unnamed			
- Functions	-	✓	$\lambda X\ X$
- Predicates	-	✓	$\lambda X\ (X = X)$
Statements about			
- Funcs/Preds/Sets/Rels	-	✓	$\mathit{reflexive} \leftrightarrow$
Powerful definitions	-	✓	$\mathit{reflexive} := \lambda R\ \forall X\ (R\ X\ X)$

Classical Higher-Order Logic (HOL)

Expressivity	FOL	HOL	Example
Quantification over			
- Individuals	✓	✓	$\forall X_i \; p_{i\to o}(f_{i\to i}(X_i))$
- Functions	-	✓	$\forall F_{i\to i} \; p_{i\to o}(F_{i\to o}(a_i))$
- Predicates/Sets/Rels	-	✓	$\forall P_{i\to o} \; P_{i\to o}(f_{i\to i}(a_i))$
Unnamed			
- Functions	-	✓	$\lambda X_i \; X_i$
- Predicates	-	✓	$\lambda X_i \; (X = X)_i$
Statements about			
- Funcs/Preds/Sets/Rels	-	✓	$reflexive_{(o\to o\to o)\to o} \leftrightarrow_{o\to o\to o}$
Powerful definitions	-	✓	$reflexive_{(\alpha\to\alpha\to o)\to o} := \lambda R_{(\alpha\to\alpha\to o)} \; \forall X_\alpha \; (R\;X\;X)$

Types: Prevent paradoxes and inconsistencies

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):

$$\boldsymbol{s, t \quad ::= \quad \underbrace{p_\alpha}_{\text{constants}} \quad \mid \quad \underbrace{X_\alpha}_{\text{variables}} \quad \mid \quad \underbrace{(\lambda X_\alpha s_\beta)_{\alpha\to\beta}}_{\text{lambda abstraction}} \quad \mid \quad \underbrace{(s_{\alpha\to\beta}\, t_\alpha)_\beta}_{\text{application}}}$$

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

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: 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):

$$\boldsymbol{s}, \boldsymbol{t} \quad ::= \quad \underbrace{\boldsymbol{p}_\alpha}_{\text{constants}} \quad \mid \quad \underbrace{\boldsymbol{X}_\alpha}_{\text{variables}} \quad \mid \quad \underbrace{(\lambda \boldsymbol{X}_\alpha \boldsymbol{s}_\beta)_{\alpha \to \beta}}_{\text{lambda abstraction}} \quad \mid \quad \underbrace{(\boldsymbol{s}_{\alpha \to \beta}\, \boldsymbol{t}_\alpha)_\beta}_{\text{application}}$$

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

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: 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.: $((\lambda X\ (p\ X))\ t) \xrightarrow{\beta-reduction} (p\ t)$

HOL defined on Top of Simply Typed λ-Calculus

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

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

- 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: 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):

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

constants variables lambda abstraction application

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

HOL defined on Top of Simply Typed λ-Calculus

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

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

- 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: Simple Syntax

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

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

Simply Typed λ-Calculus (with constants):

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

constants variables lambda abstraction application

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

HOL defined on Top of Simply Typed λ-Calculus

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

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

- no binder besides λ needed: $\forall X_\alpha\ s_o$ stands for $\Pi_{(\alpha \rightarrow o) \rightarrow o}(\lambda X_\alpha\ s_o)$
- $\bot, \top, \rightarrow, \leftrightarrow, \exists, =$ 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: 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):

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

constants variables lambda abstraction application

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

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: 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):

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

constants variables lambda abstraction application

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

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: 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):

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

constants variables lambda abstraction application

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

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: 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):

$$\boldsymbol{s}, \boldsymbol{t} \quad ::= \quad \boldsymbol{p}_\alpha \quad \mid \quad \boldsymbol{X}_\alpha \quad \mid \quad (\boldsymbol{\lambda X_\alpha s_\beta})_{\alpha\to\beta} \quad \mid \quad (\boldsymbol{s}_{\alpha\to\beta}\, \boldsymbol{t}_\alpha)_\beta$$

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: 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):

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

constants variables lambda abstraction application

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

HOL defined on Top of Simply Typed λ-Calculus

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

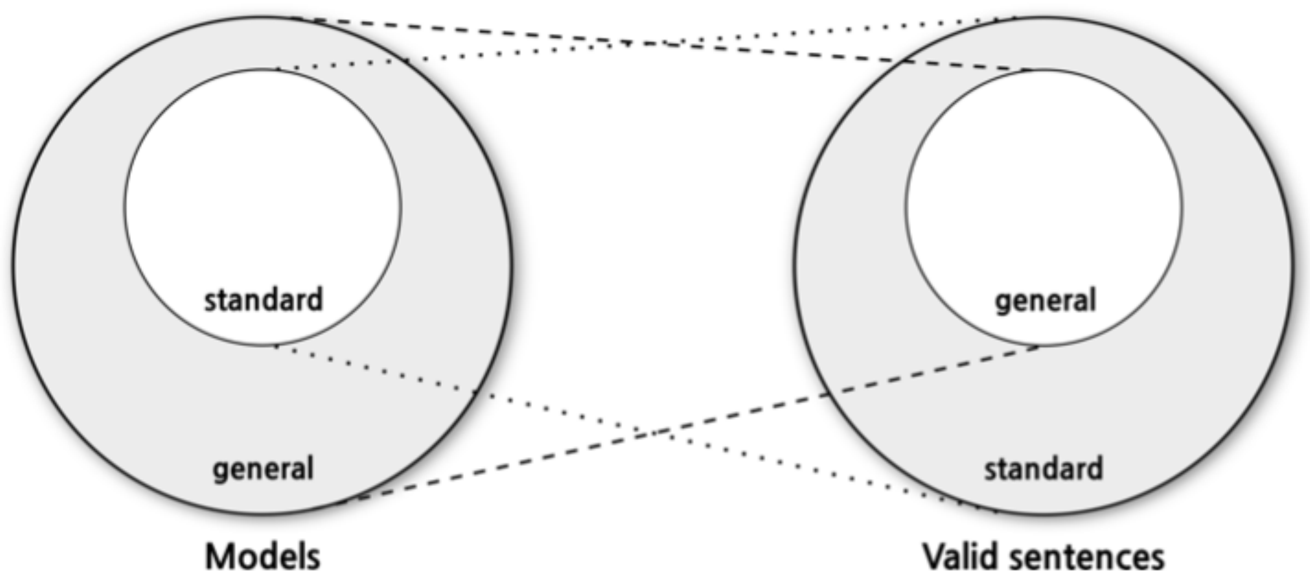
  $\neg_{o\to o} \quad \vee_{o\to o\to o} \quad \Pi_{(\alpha\to o)\to o}$ (or only $=_{\alpha\to\alpha\to o}$)

- 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: $\forall P \, \forall Q \, ((P \leftrightarrow Q) \rightarrow P = Q)$
- Functional Extensionality: $\forall F \, \forall G \, ((\forall X \, F \, X = G \, X) \rightarrow F = G)$

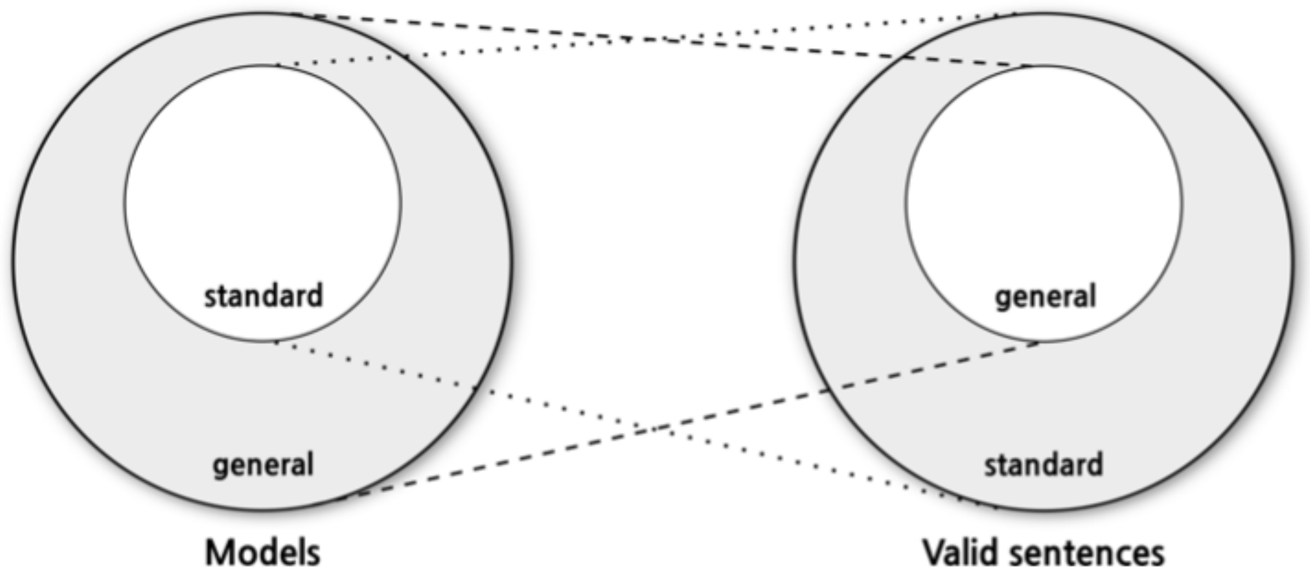
Note: Any "Henkin-valid" formula is also valid in standard semantics!

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]

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: $\forall P\, \forall Q\, ((P \leftrightarrow Q) \rightarrow P = Q)$
- Functional Extensionality: $\forall F\, \forall G\, ((\forall X\, F\, X = G\, X) \rightarrow F = G)$

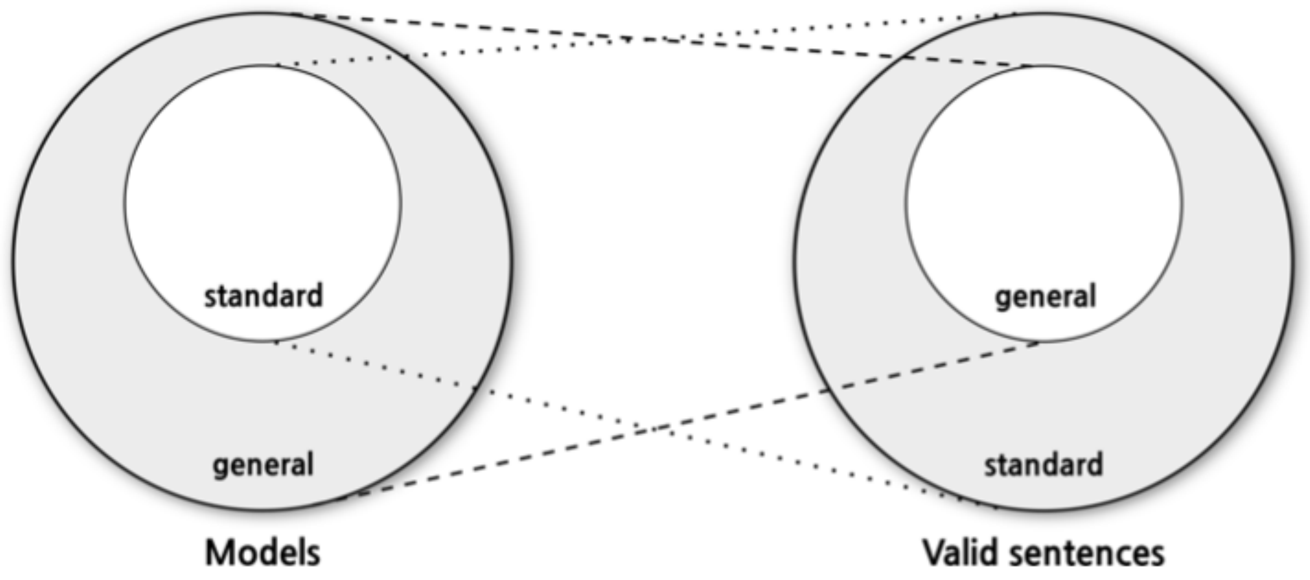
Note: Any "Henkin-valid" formula is also valid in standard semantics!

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]

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: $\forall P\, \forall Q\, ((P \leftrightarrow Q) \rightarrow P = Q)$
- Functional Extensionality: $\forall F\, \forall G\, ((\forall X\, F\; X = G\; X) \rightarrow F = G)$

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]

LogiKEy


Applications
Domain-Specific Language(s)/Theorie(s)
Object Logic(s)
Meta-Logic (HOL)
HOL (meta-logic) φ ::=
Your-logic (object-logic) ψ ::=
Embedding of in
=
=
=
=
Embedding of meta-logical notions on in
valid =
satisfiable =
... =
Pass this set of equations to a higher-order automated theorem prover

Universal Logical Reasoning

Approach: Shallow Semantic Embedding in HOL

L (target logic): $s, t ::=$

HOL (meta-logic): $s, t ::=$

Embedding of in

Signature of HOL (Constants and Logical Symbols):

$:=$

$:=$

$:=$

$:=$

Meta-logical notions:

$:=$

Formulas of L are directly identified with terms HOL

Universal Logical Reasoning

Kripke Style Semantics (propositional modal logic K)

$M, g, s \models P$ if and only if $s \in g(P)$

$M, g, s \models \neg \varphi$ if and only if $M, g, s \not\models \varphi$

$M, g, s \models \varphi \vee \psi$ if and only if $M, g, s \models \varphi$ or $M, g, s \models \psi$

$M, g, s \models \Box \varphi$ if and only if for all t with sRt we have $M, g, t \models \varphi$

Universal Logical Reasoning

Kripke Style Semantics (propositional modal logic K)

$M, g, s \models P$ if and only if $s \in g(P)$

$M, g, s \models \neg \varphi$ if and only if $M, g, s \not\models \varphi$

$M, g, s \models \varphi \vee \psi$ if and only if $M, g, s \models \varphi$ or $M, g, s \models \psi$

$M, g, s \models \Box \varphi$ if and only if for all t with sRt we have $M, g, t \models \varphi$

Standard Translation for Propositional Fragment (encoded in HOL)

- P = $P_{i \to o}$
- $\neg$ = $\lambda \varphi_{i \to o}\ \lambda w_i\ \neg(\varphi w)$
- $\vee$ = $\lambda \varphi_{i \to o}\ \lambda \psi_{i \to o}\ \lambda w_i\ \varphi w \vee \psi w$
- $\Box$ = $\lambda \varphi_{i \to o}\ \lambda w_i\ \forall v_i\ Rwv \to \varphi v$

Validity

- $[\varphi_{i \to o}] = \forall w_i\ \varphi\ w$

Modal Logic (in fact, Hybrid Logic) as a Fragment of HOL

Universal Logical Reasoning

Kripke Style Semantics (adding quantifiers)

$M, g, s \models \forall x\, \varphi$ if and only if for all $d \in D$ we have $M, ([d/x]g), s \models \varphi$

Standard Translation extended for Quantifiers (and encoded in HOL)

- remember: $\forall x_\alpha\ s$ is shorthand for $\Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ s)$ —no binder needed!!!
- $\Pi = \lambda\Phi_{\alpha\to(i\to o)}\ \lambda w_i\ \Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ \Phi x w)$

Example (compositionality and λ-conversion at work; omitting types)

$$
\begin{array}{lll}
[\Box\forall x\ Px] & \equiv\ [\Box\ \overbrace{\Pi(\lambda x\ Px)}^{\forall x\ Px}\ ] & \text{def. of } \forall \\
 & \equiv\ [\Box(\overbrace{(\lambda\Phi\ \lambda w\ \Pi(\lambda x\ \Phi x w))}^{\Pi}(\lambda x\ Px))] & \text{def. of } \Pi \\
 & \equiv\ [\Box(\lambda w\ \Pi(\lambda x\ Pxw))] & \beta\text{-reduction} \\
 & \equiv\ [\ \overbrace{(\lambda\varphi\ \lambda w\ \Pi(\lambda v\ Rwv \to \varphi v))}^{\Box}(\lambda w\ \Pi(\lambda x\ Pxw))] & \text{def. of } \Box \\
 & \equiv\ [\lambda w\ \forall v\ (Rwv \to \forall x\ Pxv)] & \beta\text{-reduction} \\
 & \equiv\ \forall w\ \forall v\ (Rwv \to \forall x\ Pxv). \quad \text{def. of validity,} & \beta\text{-reduction}
\end{array}
$$

Universal Logical Reasoning

Kripke Style Semantics (adding quantifiers)

$M, g, s \models \forall x\, \varphi$ if and only if for all $d \in D$ we have $M, ([d/x]g), s \models \varphi$

Standard Translation extended for Quantifiers (and encoded in HOL)

- remember: $\forall x_\alpha\ s$ is shorthand for $\Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ s)$ —no binder needed!!!
- $\Pi = \lambda \Phi_{\alpha\to(i\to o)}\ \lambda w_i\ \Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ \Phi x w)$

Example (compositionality and λ-conversion at work; omitting types)

$$
\begin{array}{rcll}
[\Box\forall x\, Px] & \equiv & [\Box\, \overbrace{\Pi(\lambda x\, Px)}^{\forall x\, Px}\,] & \text{def. of } \forall \\
 & \equiv & [\Box(\overbrace{(\lambda\Phi\ \lambda w\ \Pi(\lambda x\ \Phi x w))}^{\Pi}(\lambda x\ Px))] & \text{def. of } \Pi \\
 & \equiv & [\Box(\lambda w\ \Pi(\lambda x\ Pxw))] & \beta\text{-reduction} \\
 & \equiv & [\,\overbrace{(\lambda\varphi\ \lambda w\ \Pi(\lambda v\ Rwv \to \varphi v))}^{\Box}(\lambda w\ \Pi(\lambda x\ Pxw))] & \text{def. of } \Box \\
 & \equiv & [\lambda w\ \forall v\ (Rwv \to \forall x\ Pxv)] & \beta\text{-reduction} \\
 & \equiv & \forall w\ \forall v\ (Rwv \to \forall x\ Pxv). & \text{def. of validity,} \quad \beta\text{-reduction}
\end{array}
$$

Universal Logical Reasoning

Kripke Style Semantics (adding quantifiers)

$M, g, s \models \forall x\, \varphi$ if and only if for all $d \in D$ we have $M, ([d/x]g), s \models \varphi$

Standard Translation extended for Quantifiers (and encoded in HOL)

- remember: $\forall x_\alpha\ s$ is shorthand for $\Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ s)$ —no binder needed!!!
- $\Pi = \lambda \Phi_{\alpha\to(i\to o)}\ \lambda w_i\ \Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ \Phi x w)$

Example (compositionality and λ-conversion at work; omitting types)

$$
\begin{aligned}
[\Box\forall x\ Px] &\equiv [\Box\ \overbrace{\Pi(\lambda x\ Px)}^{\forall x\ Px}\ ] && \text{def. of } \forall \\
&\equiv [\Box(\overbrace{(\lambda\Phi\ \lambda w\ \Pi(\lambda x\ \Phi x w))}^{\Pi}(\lambda x\ Px))] && \text{def. of } \Pi \\
&\equiv [\Box(\lambda w\ \Pi(\lambda x\ Pxw))] && \beta\text{-reduction} \\
&\equiv [\ \overbrace{(\lambda\varphi\ \lambda w\ \Pi(\lambda v\ Rwv \to \varphi v))}^{\Box}(\lambda w\ \Pi(\lambda x\ Pxw))] && \text{def. of } \Box \\
&\equiv [\lambda w\ \forall v\ (Rwv \to \forall x\ Pxv)] && \beta\text{-reduction} \\
&\equiv \forall w\ \forall v\ (Rwv \to \forall x\ Pxv). && \text{def. of validity,} \quad \beta\text{-reduction}
\end{aligned}
$$

Universal Logical Reasoning

Kripke Style Semantics (adding quantifiers)

$M, g, s \models \forall x\, \varphi$ if and only if for all $d \in D$ we have $M, ([d/x]g), s \models \varphi$

Standard Translation extended for Quantifiers (and encoded in HOL)

- remember: $\forall x_\alpha\ s$ is shorthand for $\Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ s)$ —no binder needed!!!
- $\Pi = \lambda \Phi_{\alpha\to(i\to o)}\ \lambda w_i\ \Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ \Phi x w)$

Example (compositionality and λ-conversion at work; omitting types)

$$
\begin{aligned}
[\Box \forall x\ Px] &\equiv [\Box\ \overbrace{\Pi(\lambda x\ Px)}^{\forall x\ Px}\ ] && \text{def. of } \forall \\
&\equiv [\Box(\overbrace{(\lambda \Phi\ \lambda w\ \Pi(\lambda x\ \Phi x w))}^{\Pi}(\lambda x\ Px))] && \text{def. of } \Pi \\
&\equiv [\Box(\lambda w\ \Pi(\lambda x\ Pxw))] && \beta\text{-reduction} \\
&\equiv [\ \overbrace{(\lambda \varphi\ \lambda w\ \Pi(\lambda v\ Rwv \to \varphi v))}^{\Box}(\lambda w\ \Pi(\lambda x\ Pxw))] && \text{def. of } \Box \\
&\equiv [\lambda w\ \forall v\ (Rwv \to \forall x\ Pxv)] && \beta\text{-reduction} \\
&\equiv \forall w\ \forall v\ (Rwv \to \forall x\ Pxv). && \text{def. of validity,}\quad \beta\text{-reduction}
\end{aligned}
$$

Universal Logical Reasoning

Kripke Style Semantics (adding quantifiers)

$M, g, s \models \forall x\, \varphi$ if and only if for all $d \in D$ we have $M, ([d/x]g), s \models \varphi$

Standard Translation extended for Quantifiers (and encoded in HOL)

- remember: $\forall x_\alpha\ s$ is shorthand for $\Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ s)$ —no binder needed!!!
- $\Pi = \lambda \Phi_{\alpha\to(i\to o)}\ \lambda w_i\ \Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ \Phi x w)$

Example (compositionality and λ-conversion at work; omitting types)

$$
\begin{array}{rlll}
[\Box\forall x\ Px] & \equiv & [\Box\, \overbrace{\Pi(\lambda x\ Px)}^{\forall x\ Px}\,] & \text{def. of } \forall \\
 & \equiv & [\Box(\overbrace{(\lambda\Phi\ \lambda w\ \Pi(\lambda x\ \Phi x w))}^{\Pi}(\lambda x\ Px))] & \text{def. of } \Pi \\
 & \equiv & [\Box(\lambda w\ \Pi(\lambda x\ Pxw))] & \beta\text{-reduction} \\
 & \equiv & [\,\overbrace{(\lambda\varphi\ \lambda w\ \Pi(\lambda v\ Rwv \to \varphi v))}^{\Box}(\lambda w\ \Pi(\lambda x\ Pxw))] & \text{def. of } \Box \\
 & \equiv & [\lambda w\ \forall v\ (Rwv \to \forall x\ Pxv)] & \beta\text{-reduction} \\
 & \equiv & \forall w\ \forall v\ (Rwv \to \forall x\ Pxv). \quad \text{def. of validity,} & \beta\text{-reduction}
\end{array}
$$

Universal Logical Reasoning

Kripke Style Semantics (adding quantifiers)

$M, g, s \models \forall x\, \varphi$ if and only if for all $d \in D$ we have $M, ([d/x]g), s \models \varphi$

Standard Translation extended for Quantifiers (and encoded in HOL)

- remember: $\forall x_\alpha\ s$ is shorthand for $\Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ s)$ —no binder needed!!!
- $\Pi = \lambda\Phi_{\alpha\to(i\to o)}\ \lambda w_i\ \Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ \Phi x w)$

Example (compositionality and λ-conversion at work; omitting types)

$$
\begin{aligned}
[\Box\forall x\ Px] &\equiv [\Box\ \overbrace{\Pi(\lambda x\ Px)}^{\forall x\ Px}\ ] && \text{def. of } \forall \\
&\equiv [\Box(\overbrace{(\lambda\Phi\ \lambda w\ \Pi(\lambda x\ \Phi x w))}^{\Pi}(\lambda x\ Px))] && \text{def. of } \Pi \\
&\equiv [\Box(\lambda w\ \Pi(\lambda x\ Pxw))] && \beta\text{-reduction} \\
&\equiv [\ \overbrace{(\lambda\varphi\ \lambda w\ \Pi(\lambda v\ Rwv \to \varphi v))}^{\Box}(\lambda w\ \Pi(\lambda x\ Pxw))] && \text{def. of } \Box \\
&\equiv [\lambda w\ \forall v\ (Rwv \to \forall x\ Pxv)] && \beta\text{-reduction} \\
&\equiv \forall w\ \forall v\ (Rwv \to \forall x\ Pxv). && \text{def. of validity, } \beta\text{-reduction}
\end{aligned}
$$

Universal Logical Reasoning

Kripke Style Semantics (adding quantifiers)

$M, g, s \models \forall x\, \varphi$ if and only if for all $d \in D$ we have $M, ([d/x]g), s \models \varphi$

Standard Translation extended for Quantifiers (and encoded in HOL)

- remember: $\forall x_\alpha\ s$ is shorthand for $\Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ s)$ —no binder needed!!!
- $\Pi = \lambda \Phi_{\alpha\to(i\to o)}\ \lambda w_i\ \Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ \Phi x w)$

Example (compositionality and λ-conversion at work; omitting types)

$$
\begin{array}{lcll}
[\Box\forall x\ Px] & \equiv & [\Box\ \overbrace{\Pi(\lambda x\ Px)}^{\forall x\ Px}\ ] & \text{def. of } \forall \\
 & \equiv & [\Box(\overbrace{(\lambda\Phi\ \lambda w\ \Pi(\lambda x\ \Phi x w))}^{\Pi}(\lambda x\ Px))] & \text{def. of } \Pi \\
 & \equiv & [\Box(\lambda w\ \Pi(\lambda x\ Pxw))] & \beta\text{-reduction} \\
 & \equiv & [\ \overbrace{(\lambda\varphi\ \lambda w\ \Pi(\lambda v\ Rwv \to \varphi v))}^{\Box}(\lambda w\ \Pi(\lambda x\ Pxw))] & \text{def. of } \Box \\
 & \equiv & [\lambda w\ \forall v\ (Rwv \to \forall x\ Pxv)] & \beta\text{-reduction} \\
 & \equiv & \forall w\ \forall v\ (Rwv \to \forall x\ Pxv). & \text{def. of validity,}\quad \beta\text{-reduction}
\end{array}
$$

Universal Logical Reasoning

Kripke Style Semantics (adding quantifiers)

$M, g, s \models \forall x\, \varphi$ if and only if for all $d \in D$ we have $M, ([d/x]g), s \models \varphi$

Standard Translation extended for Quantifiers (and encoded in HOL)

- remember: $\forall x_\alpha\ s$ is shorthand for $\Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ s)$ —no binder needed!!!
- $\Pi = \lambda \Phi_{\alpha\to(i\to o)}\ \lambda w_i\ \Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ \Phi x w)$

Example (compositionality and λ-conversion at work; omitting types)

$$
\begin{array}{lcll}
[\Box\forall x\ Px] & \equiv & [\Box\ \overbrace{\Pi(\lambda x\ Px)}^{\forall x\ Px}\ ] & \text{def. of } \forall \\
 & \equiv & [\Box(\overbrace{(\lambda \Phi\ \lambda w\ \Pi(\lambda x\ \Phi x w))}^{\Pi}(\lambda x\ Px))] & \text{def. of } \Pi \\
 & \equiv & [\Box(\lambda w\ \Pi(\lambda x\ Pxw))] & \beta\text{-reduction} \\
 & \equiv & [\ \overbrace{(\lambda \varphi\ \lambda w\ \Pi(\lambda v\ Rwv \to \varphi v))}^{\Box}(\lambda w\ \Pi(\lambda x\ Pxw))] & \text{def. of } \Box \\
 & \equiv & [\lambda w\ \forall v\ (Rwv \to \forall x\ Pxv)] & \beta\text{-reduction} \\
 & \equiv & \forall w\ \forall v\ (Rwv \to \forall x\ Pxv). \quad \text{def. of validity,} & \beta\text{-reduction}
\end{array}
$$

Universal Logical Reasoning

Kripke Style Semantics (adding quantifiers)

$M, g, s \models \forall x\, \varphi$ if and only if for all $d \in D$ we have $M, ([d/x]g), s \models \varphi$

Standard Translation extended for Quantifiers (and encoded in HOL)

- remember: $\forall x_\alpha\ s$ is shorthand for $\Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ s)$ —no binder needed!!!
- $\Pi = \lambda\Phi_{\alpha\to(i\to o)}\ \lambda w_i\ \Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ \Phi x w)$

Example (compositionality and λ-conversion at work; omitting types)

$$
\begin{array}{lcll}
[\Box\forall x\ Px] & \equiv & [\Box\, \overbrace{\Pi(\lambda x\ Px)}^{\forall x\ Px}\,] & \text{def. of } \forall \\
 & \equiv & [\Box(\overbrace{(\lambda\Phi\ \lambda w\ \Pi(\lambda x\ \Phi x w))}^{\Pi}(\lambda x\ Px))] & \text{def. of } \Pi \\
 & \equiv & [\Box(\lambda w\ \Pi(\lambda x\ Pxw))] & \beta\text{-reduction} \\
 & \equiv & [\,\overbrace{(\lambda\varphi\ \lambda w\ \Pi(\lambda v\ Rwv \to \varphi v))}^{\Box}(\lambda w\ \Pi(\lambda x\ Pxw))] & \text{def. of } \Box \\
 & \equiv & [\lambda w\ \forall v\ (Rwv \to \forall x\ Pxv)] & \beta\text{-reduction} \\
 & \equiv & \forall w\ \forall v\ (Rwv \to \forall x\ Pxv). & \text{def. of validity, } \beta\text{-reduction}
\end{array}
$$

Universal Logical Reasoning

Kripke Style Semantics (adding quantifiers)

$M, g, s \models \forall x\, \varphi$ if and only if for all $d \in D$ we have $M, ([d/x]g), s \models \varphi$

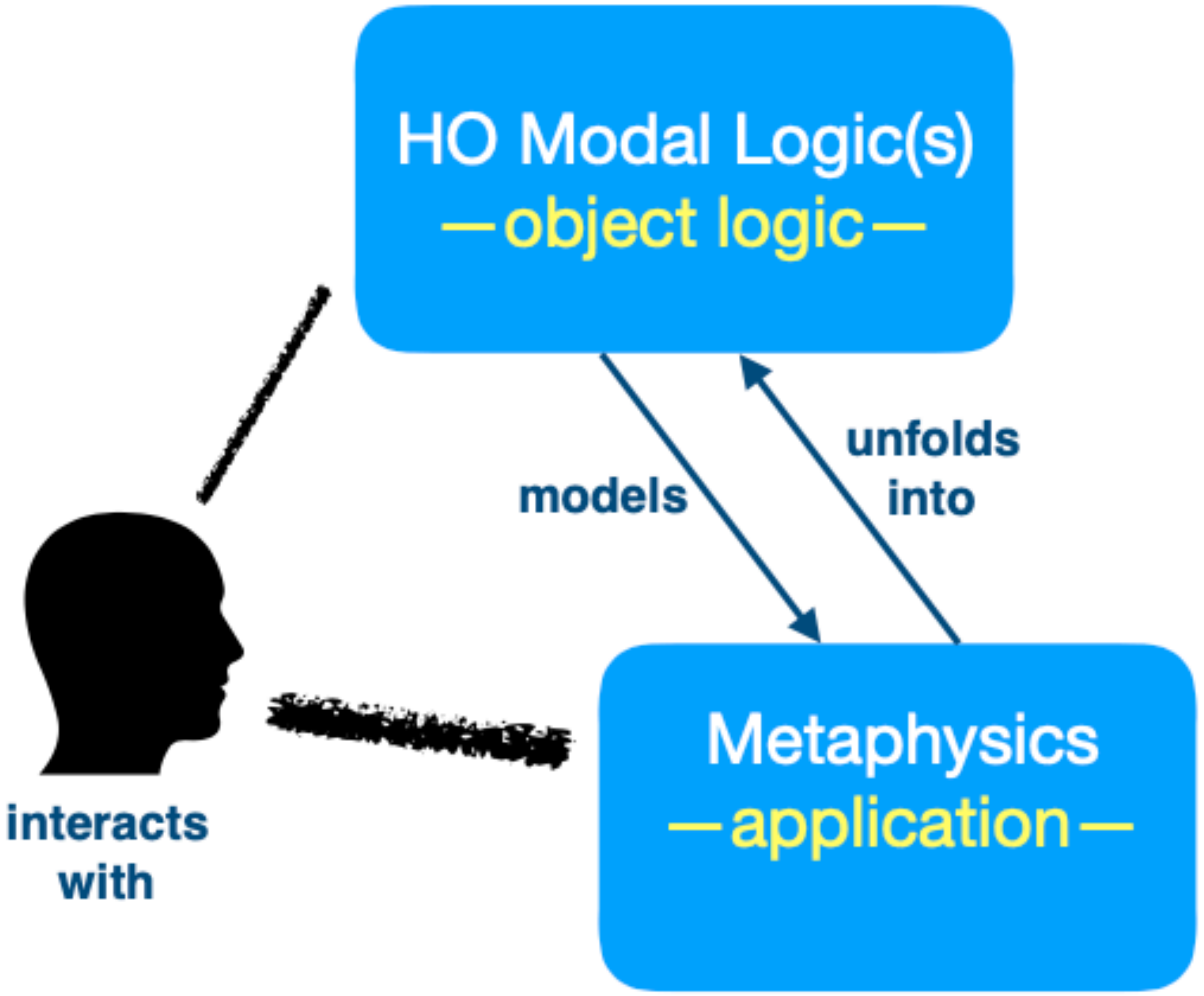
Standard Translation extended for Quantifiers (and encoded in HOL)

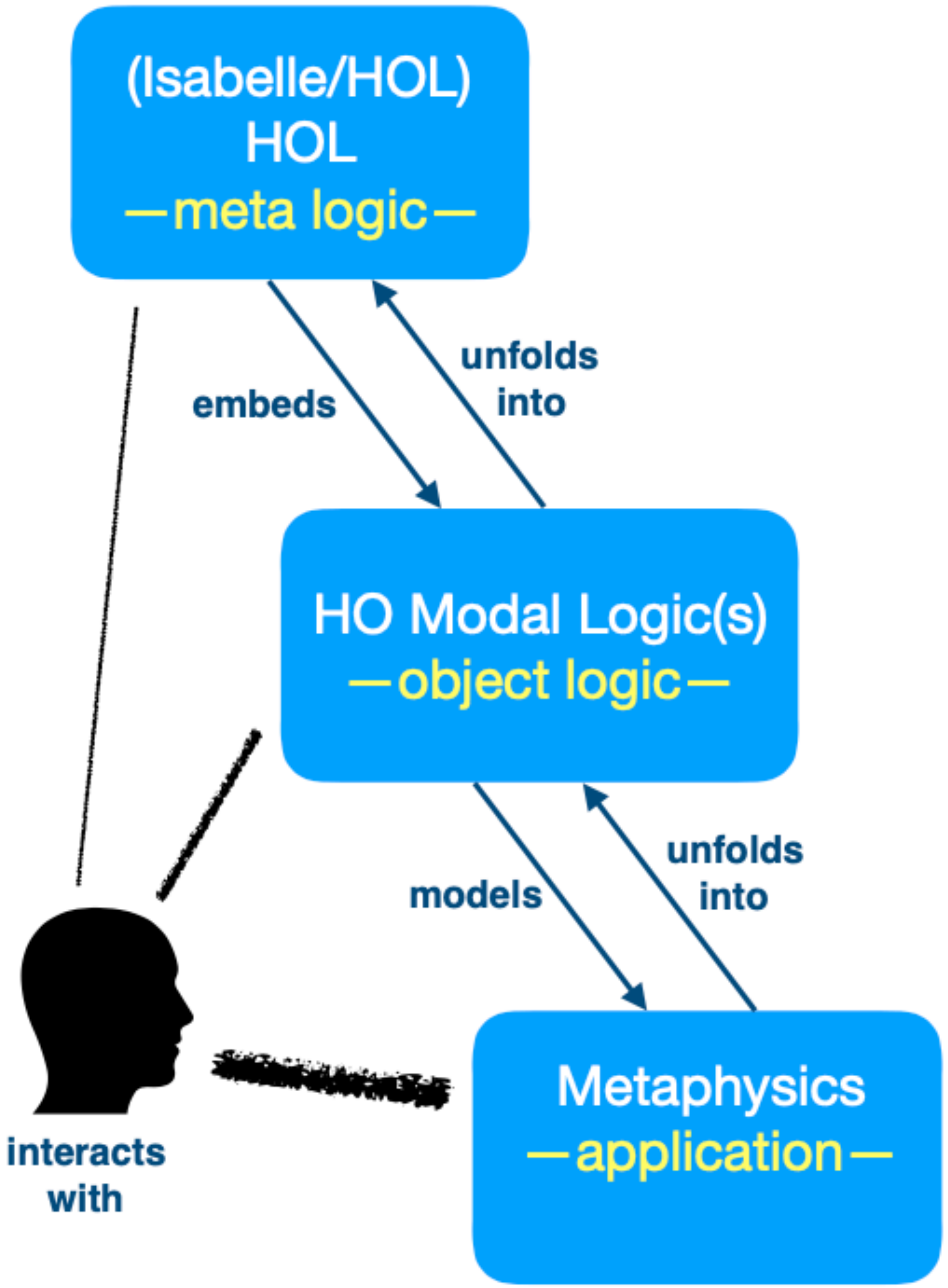
- remember: $\forall x_\alpha\ s$ is shorthand for $\Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ s)$ —no binder needed!!!
- $\Pi = \lambda\Phi_{\alpha\to(i\to o)}\ \lambda w_i\ \Pi_{(\alpha\to o)\to o}(\lambda x_\alpha\ \Phi x w)$

Example (compositionality and λ-conversion at work; omitting types)

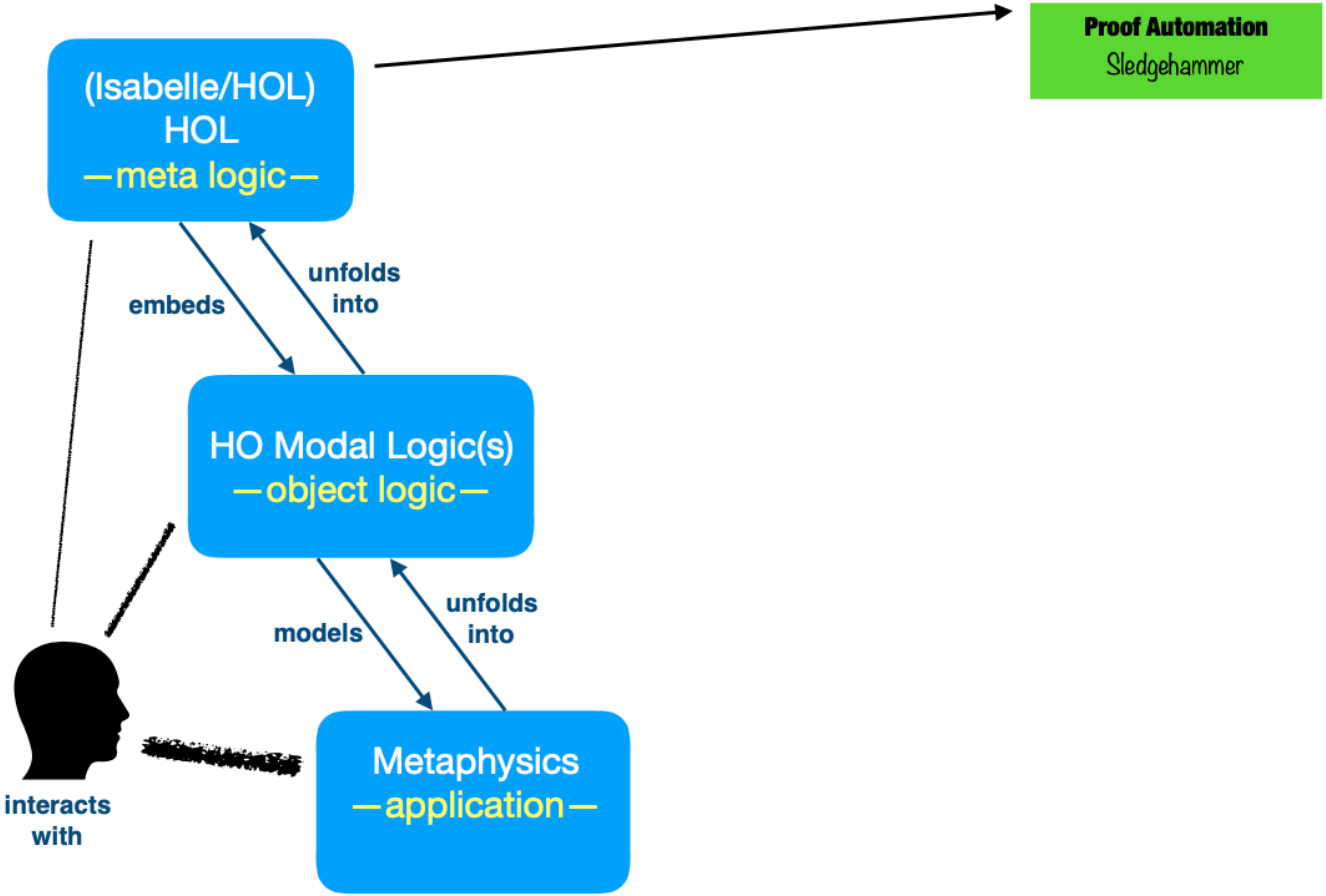
$$
\begin{array}{lcll}
[\Box\forall x\ Px] & \equiv & [\Box\, \overbrace{\Pi(\lambda x\ Px)}^{\forall x\ Px}\,] & \text{def. of } \forall \\
 & \equiv & [\Box(\overbrace{(\lambda\Phi\ \lambda w\ \Pi(\lambda x\ \Phi x w))}^{\Pi}(\lambda x\ Px))] & \text{def. of } \Pi \\
 & \equiv & [\Box(\lambda w\ \Pi(\lambda x\ Pxw))] & \beta\text{-reduction} \\
 & \equiv & [\,\overbrace{(\lambda\varphi\ \lambda w\ \Pi(\lambda v\ Rwv \to \varphi v))}^{\Box}(\lambda w\ \Pi(\lambda x\ Pxw))] & \text{def. of } \Box \\
 & \equiv & [\lambda w\ \forall v\ (Rwv \to \forall x\ Pxv)] & \beta\text{-reduction} \\
 & \equiv & \forall w\ \forall v\ (Rwv \to \forall x\ Pxv). & \text{def. of validity, } \quad \beta\text{-reduction}
\end{array}
$$


interacts
with
Metaphysics
—application—

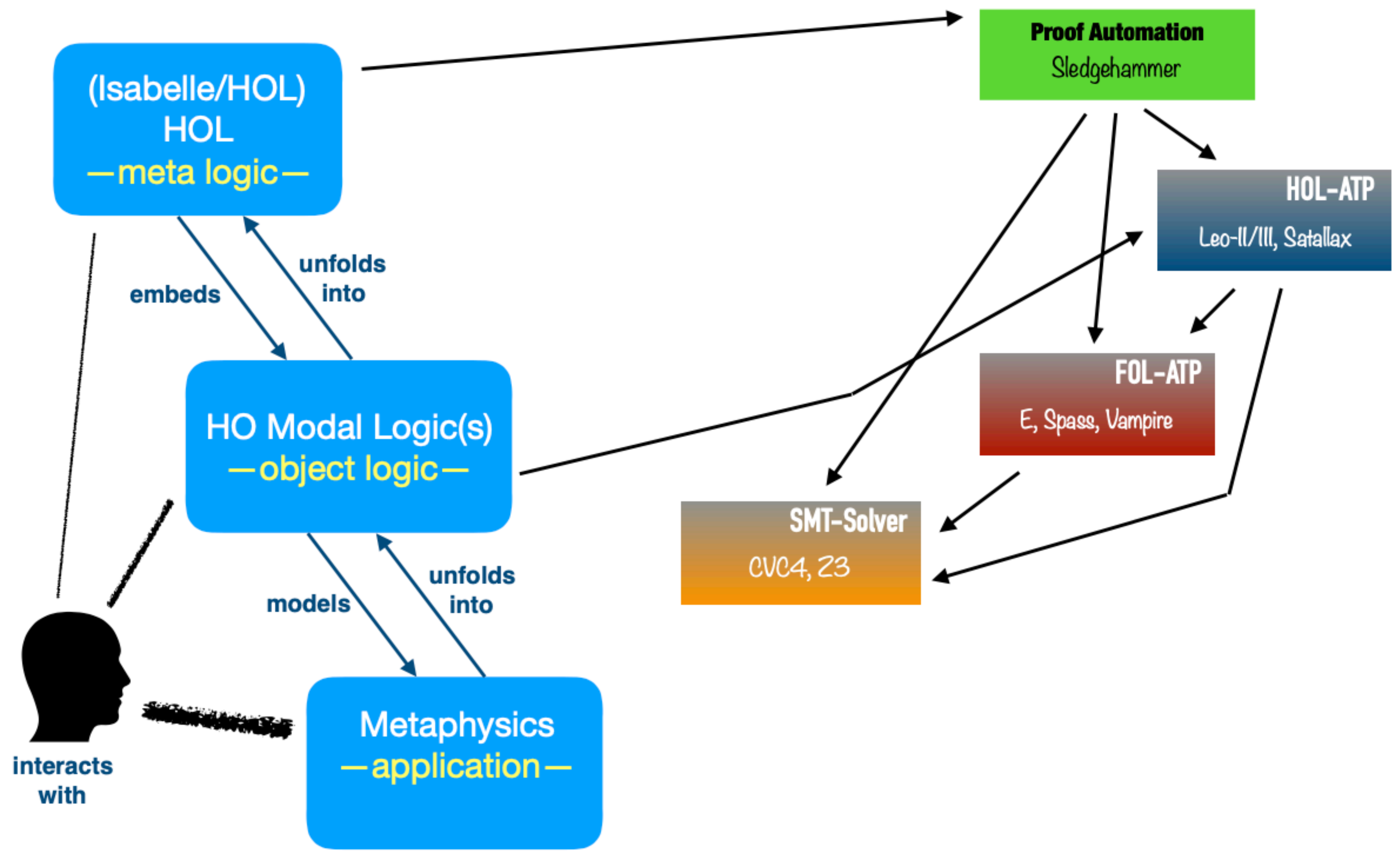

HO Modal Logic(s)
—object logic—
models
unfolds
into
Metaphysics
—application—
interacts
with

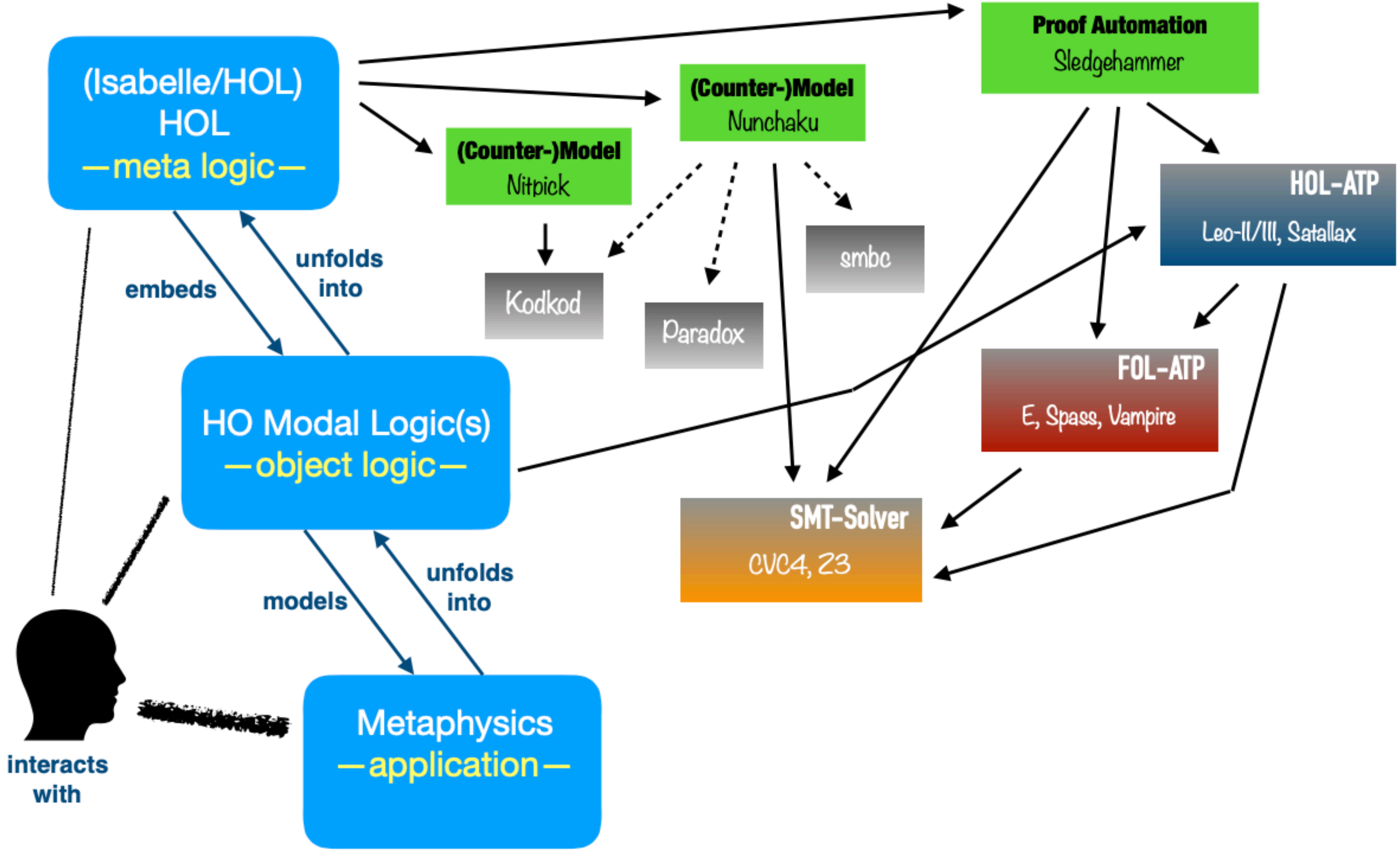


(Isabelle/HOL)
HOL
—meta logic—
embeds
unfolds into
HO Modal Logic(s)
—object logic—
models
unfolds into
Metaphysics
—application—
interacts with

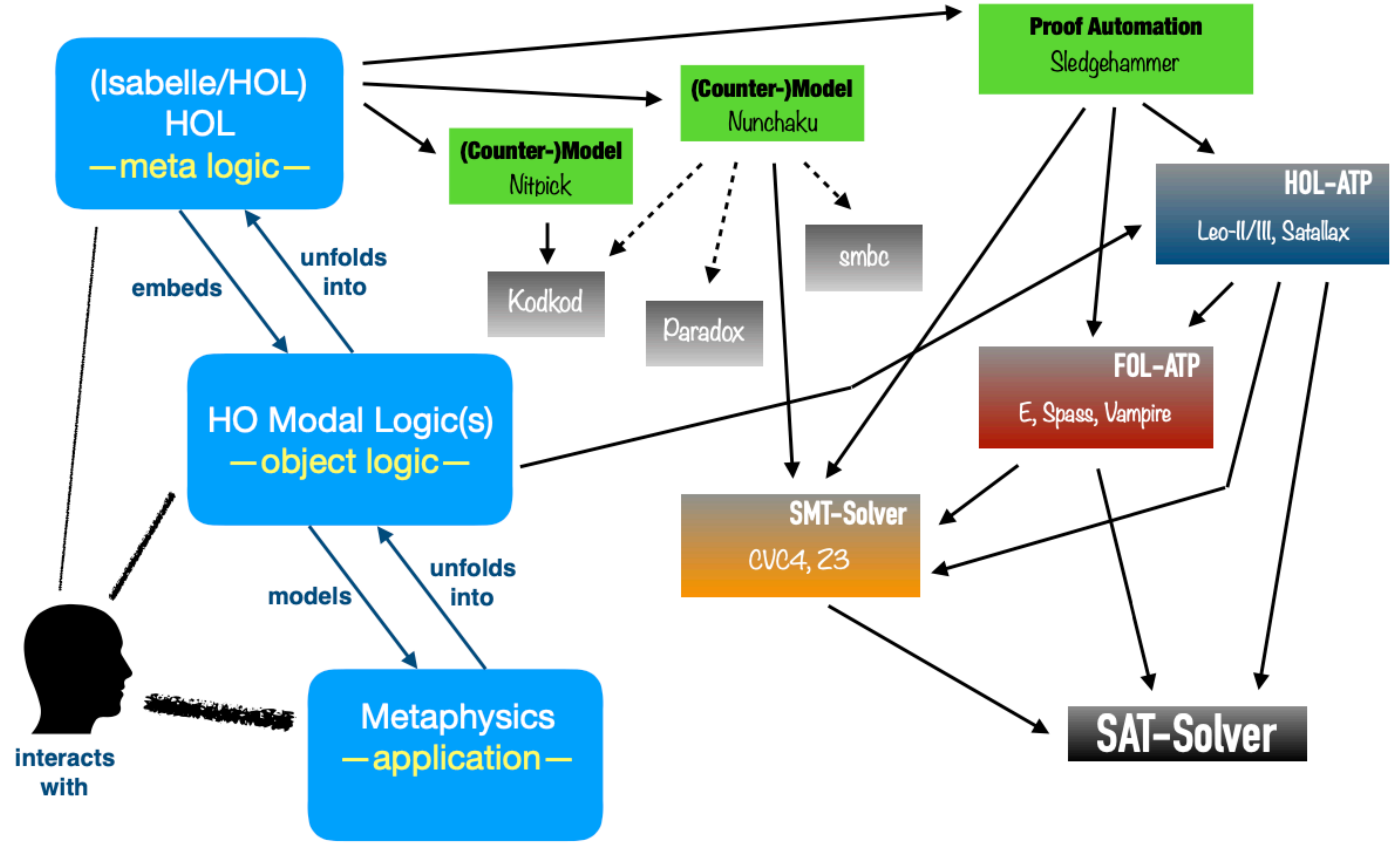


Proof Automation
Sledgehammer
(Isabelle/HOL)
HOL
—meta logic—
embeds
unfolds into
HO Modal Logic(s)
—object logic—
models
unfolds into
Metaphysics
—application—
interacts with

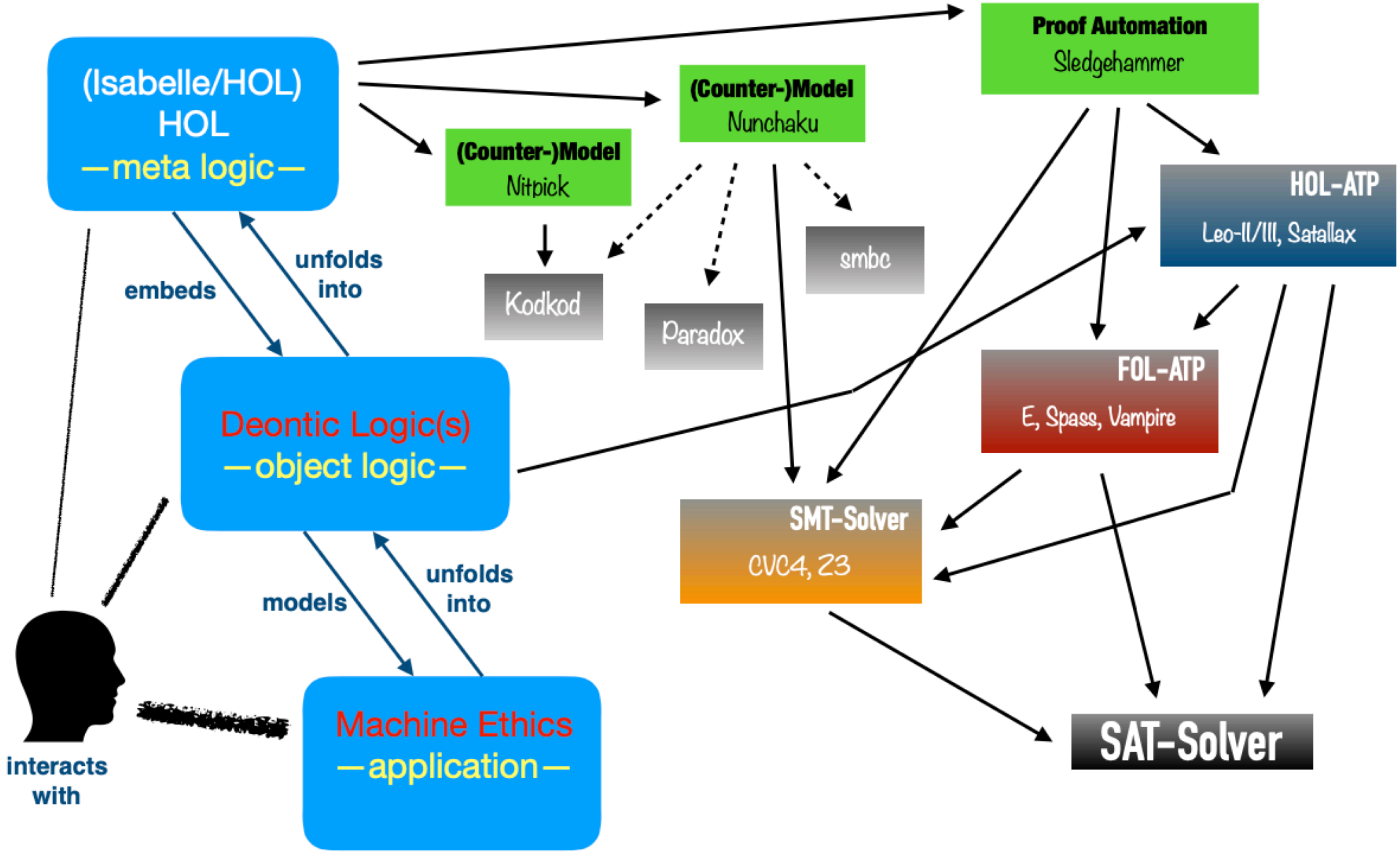

(Isabelle/HOL)
HOL
—meta logic—
embeds
unfolds into
HO Modal Logic(s)
—object logic—
models
unfolds into
Metaphysics
—application—
interacts with
Proof Automation
Sledgehammer
HOL-ATP
Leo-II/III, Satallax
FOL-ATP
E, Spass, Vampire
SMT-Solver
CVC4, Z3



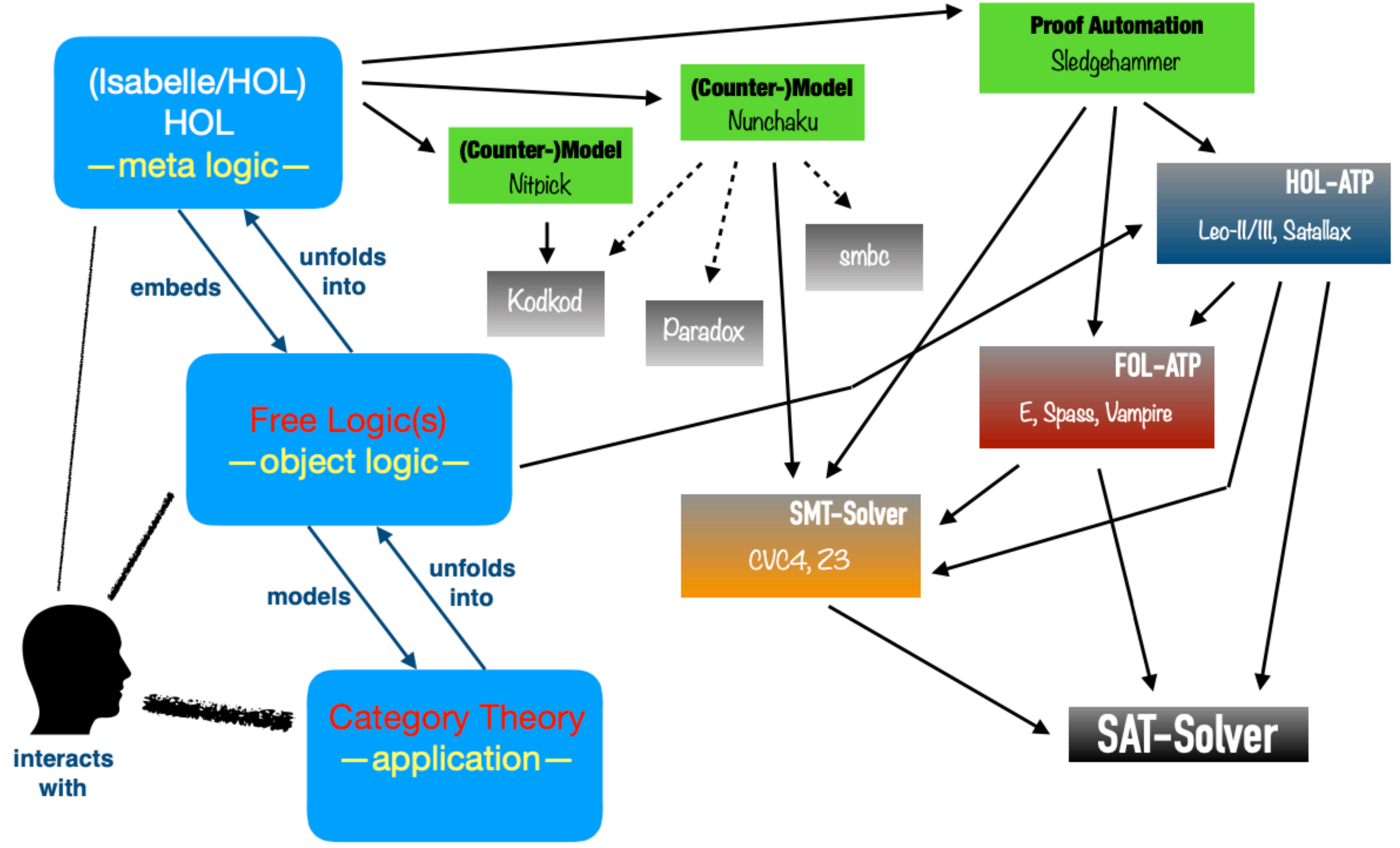
(Isabelle/HOL)
HOL
—meta logic—
embeds
unfolds into
HO Modal Logic(s)
—object logic—
models
unfolds into
Metaphysics
—application—
interacts with
(Counter-)Model
Nitpick
(Counter-)Model
Nunchaku
Proof Automation
Sledgehammer
Kodkod
Paradox
smbc
HOL-ATP
Leo-II/III, Satallax
FOL-ATP
E, Spass, Vampire
SMT-Solver
CVC4, Z3



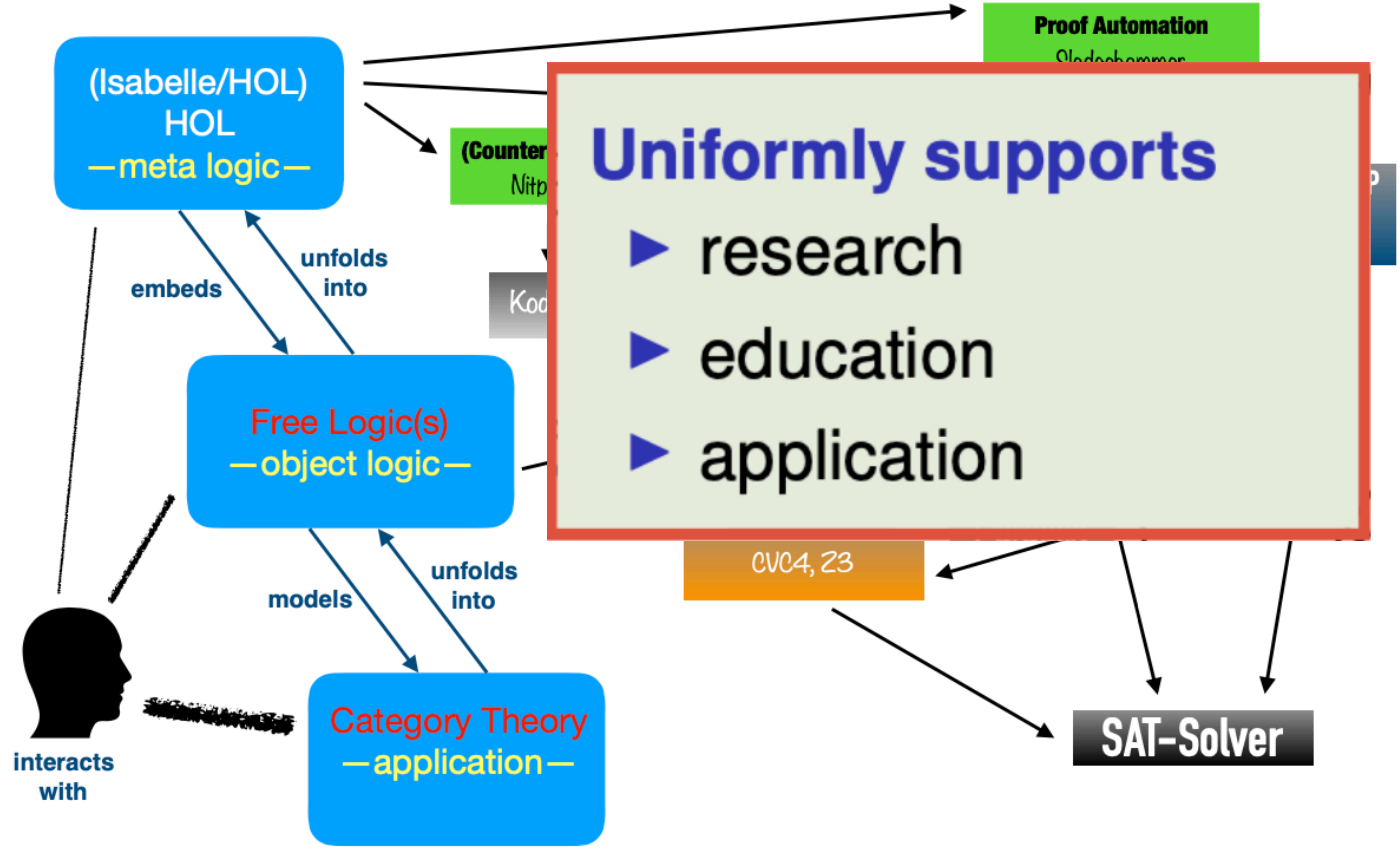
(Isabelle/HOL)
HOL
—meta logic—
Proof Automation
Sledgehammer
(Counter-)Model
Nunchaku
(Counter-)Model
Nitpick
HOL-ATP
Leo-II/III, Satallax
unfolds into
embeds
Kodkod
Paradox
smbc
FOL-ATP
E, Spass, Vampire
HO Modal Logic(s)
—object logic—
SMT-Solver
CVC4, Z3
models
unfolds into
interacts with
Metaphysics
—application—
SAT-Solver



(Isabelle/HOL)
HOL
—meta logic—
embeds
unfolds into
Deontic Logic(s)
—object logic—
models
unfolds into
Machine Ethics
—application—
interacts with
(Counter-)Model
Nitpick
(Counter-)Model
Nunchaku
Proof Automation
Sledgehammer
Kodkod
Paradox
smbc
HOL-ATP
Leo-II/III, Satallax
FOL-ATP
E, Spass, Vampire
SMT-Solver
CVC4, Z3
SAT-Solver

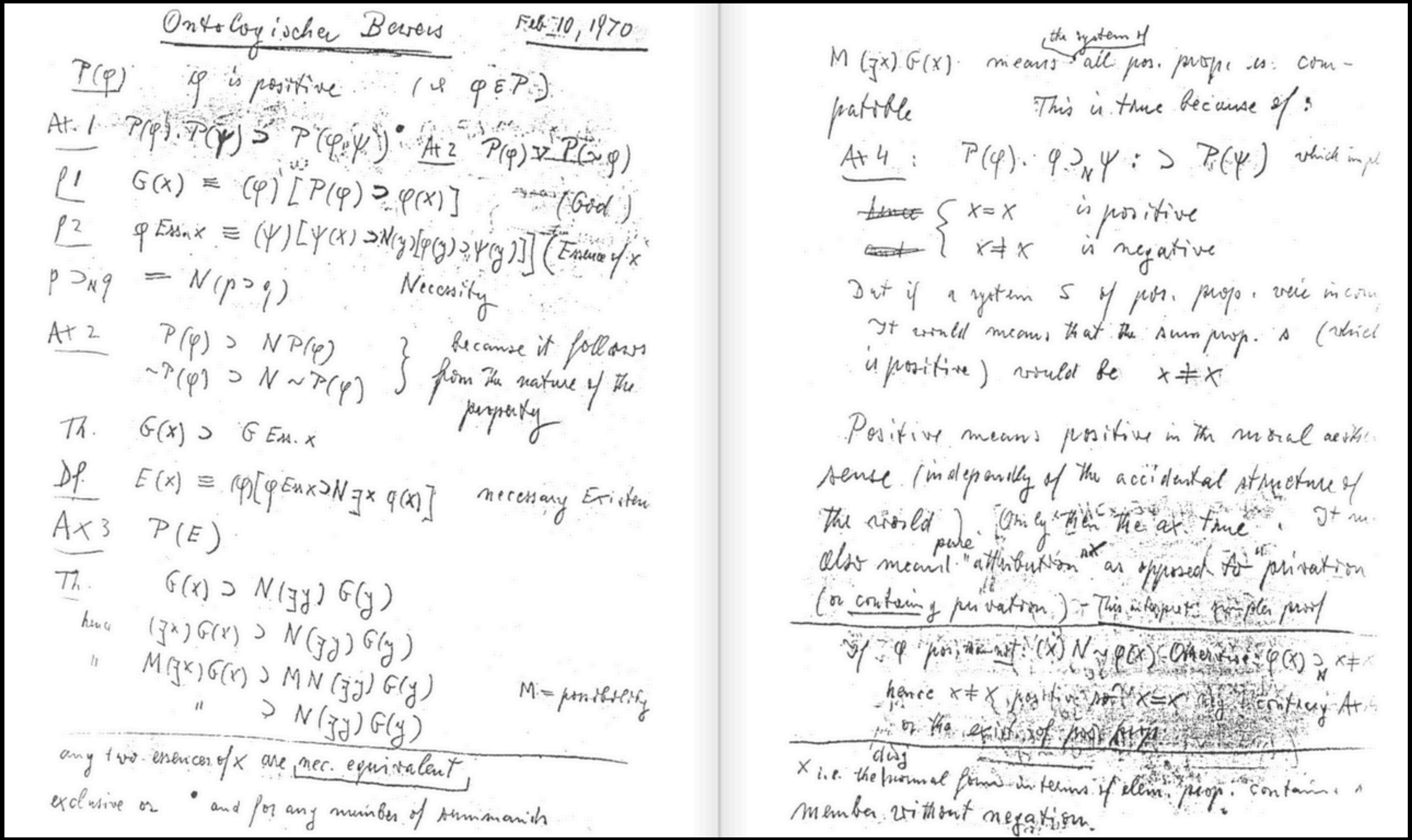


(Isabelle/HOL)
HOL
—meta logic—
(Counter-)Model
Nitpick
(Counter-)Model
Nunchaku
Proof Automation
Sledgehammer
Kodkod
Paradox
smbc
HOL-ATP
Leo-II/III, Satallax
FOL-ATP
E, Spass, Vampire
SMT-Solver
CVC4, Z3
SAT-Solver
embeds
unfolds into
Free Logic(s)
—object logic—
models
unfolds into
Category Theory
—application—
interacts with


Proof Automation
(Isabelle/HOL)
HOL
—meta logic—
embeds
unfolds into
Free Logic(s)
—object logic—
models
unfolds into
Category Theory
—application—
interacts with
CVC4, Z3
SAT-Solver
Uniformly supports
▶ research
▶ education
▶ application

Kurt Gödel

Dana Scott



Ontologischer Beweis   Feb 10, 1970

$P(\varphi)$ φ is positive ($\equiv \varphi \in P$)

At. 1 $P(\varphi).P(\psi) \supset P(\varphi.\psi)$ * At 2 $P(\varphi) \vee P(\sim\varphi)$

P1 $G(x) \equiv (\varphi)[P(\varphi) \supset \varphi(x)]$ (God)

P2 $\varphi \text{ Ess.} x \equiv (\psi)[\psi(x) \supset N(y)[\varphi(y) \supset \psi(y)]]$ (Essence of x)

$p \supset_N q = N(p \supset q)$ Necessity

At 2 $P(\varphi) \supset NP(\varphi)$; $\sim P(\varphi) \supset N \sim P(\varphi)$ } because it follows from the nature of the property

Th. $G(x) \supset G \text{ Ess.} x$

Df. $E(x) \equiv (\varphi)[\varphi \text{ Ess } x \supset N \exists x\, \varphi(x)]$ necessary Existence

Ax 3 $P(E)$

Th. $G(x) \supset N(\exists y)\, G(y)$

hence $(\exists x) G(x) \supset N(\exists y)\, G(y)$

" $M(\exists x) G(x) \supset MN(\exists y)\, G(y)$

" $\supset N(\exists y)\, G(y)$ M = possibility

any two essences of x are nec. equivalent

exclusive or * and for any number of summands

$M(\exists x)G(x)$ means the system of all pos. prop. is compatible. This is true because of:

At 4: $P(\varphi).\varphi \supset_N \psi : \supset P(\psi)$ which impl

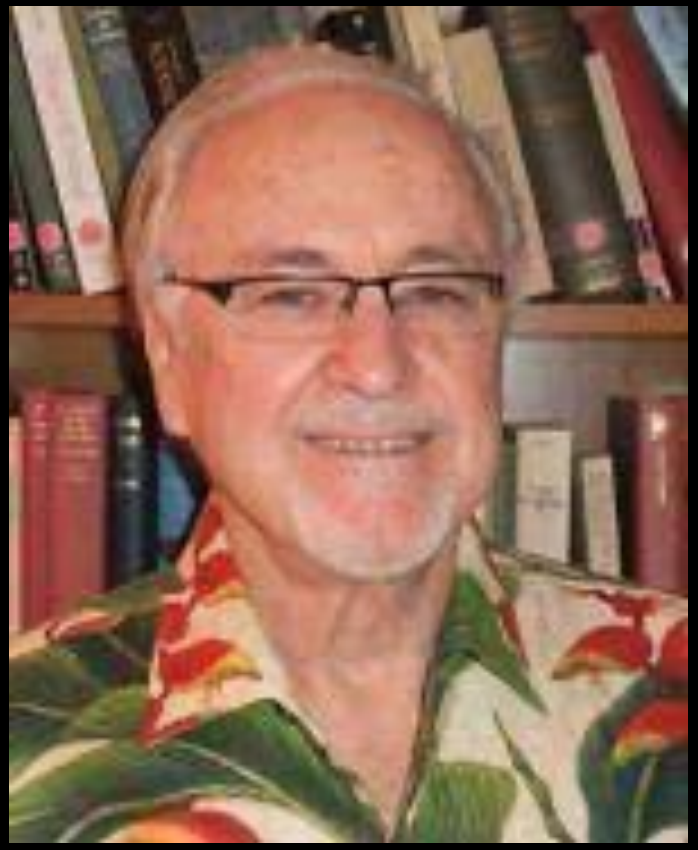
{ $x = x$ is positive; $x \neq x$ is negative

But if a system S of pos. prop. were incom it would mean that the sum prop. s (which is positive) would be $x \neq x$

Positive means positive in the moral aesth. sense (independently of the accidental structure of the world). Only then the ax. true. It m also means pure "attribution"x as opposed to "privation" (or containing privation). This interpret. simpl. proof

If φ pos. then not: $(x) N \sim \varphi(x)$. Otherwise: $\varphi(x) \supset_N x \neq x$ hence $x \neq x$ positive not $x = x$ contrary Ax or the exist. of pos. prop. [illegible]

x i.e. the disj. normal form in terms of elem. prop. contains a member without negation.



Anthony C. Anderson

Melvin Fitting

Kurt Gödel's Ontological Argument (in higher-order modal logic)

Why is all this relevant for this talk?

Talk at Berkley where Dana Scott was present:
A Success Story of Higher-Order Theorem Proving in Computational Metaphysics,
Logic Colloquium, University of California, Berkeley, USA, 2016.

$$\Pi = \lambda\phi\, \lambda w\, \forall x\, (\phi\, x w)$$ (constant domain/possibilist quantifier)

alternatively becomes

$$\Pi = \lambda\phi\, \lambda w\, \forall x\, (\underline{\text{ExistsInW}}\, x w \longrightarrow \phi\, x w)$$ (varying domain/actualist quantifier)

Dana then pointed me to free logic and suggested joint exploration studies …

(For experiments on different quantifiers see the new AFP entry at: https://www.isa-afp.org/entries/Notes_On_Goedels_Ontological_Argument.html)

Free first-order logic in HOL

Free logic in HOL

Scott 1967

Dana Scott. "Existence and description in formal logic." In: Bertrand Russell: Philosopher of the Century, edited by R. Schoenman. George Allen & Unwin, London, 1967, pp. 181-200. Reprinted with additions in: Philosophical Application of Free Logic, edited by K. Lambert. Oxford Universitry Press, 1991, pp. 28 - 48.

16

DANA SCOTT

Existence and Description in Formal Logic

The problem of what to do with improper descriptive phrases has bothered logicians for a long time. There have been three major suggestions of how to treat descriptions usually associated with the names of Russell, Frege and Hilbert-Bernays. The author does not consider any of these approaches really satisfactory. In many ways Russell's idea is most attractive because of its simplicity. However, on s

of

Technically the idea is to permit a wider interpretation of *free* variables. All bound variables retain their usual existential import (when we say something exists it does exist), but free variables behave in a more "schematic" way. Thus there will be no restrictions on the use of *modus ponens* or on the rule of *substitution* involving free variables and their occurrences. The laws of quantifiers require some modification, however, to make the existential assumptions explicit. The modification is very straightforward, and I shall argue that what has to be done is simply what is done naturally in making a *relativization* of quantifiers from a larger domain to a subdomain. Again in intuitionstic logic we have to take care over relativization, because we cannot say that either the subdomain is empty or not - thus a given element may be only "partially" in the subdomain.

IDENTITY AND EXISTENCE IN INTUITIONISTIC LOGIC

Scott 1977

Dana Scott

Merton College, Oxford, England

Standard formulations of intuitionistic

category theorists, generally do not take in

(For a recent reference see Makkai and Reyes

there is a simple psychological reason: we d

ist. Certainly we shoul

on explicit. In classic

possible to split the d

question does or does n

s, and the circumstance

ns, for example. Many p

ocate in a mild way in this paper what I consider a simple

al formulation of logic allowing reference to partial elements.

be entirely formal here, but for the model theory of the system

nsult Fourman and Scott [10] for interpretations over a complete

this includes the so-called Kripke models) and Fourman [8]

en in 1975) for the interpretation in an arbitrary topos.

(1) $Ex \leftrightarrow Edom(x)$

(2) $Ex \leftrightarrow Ecod(x)$

(3) $E(x \circ y) \leftrightarrow dom(x) = cod(y)$

(4) $x \circ (y \circ z) \equiv (x \circ y) \circ z$

(5) $x \circ dom(x) \equiv x$

(6) $cod(x) \circ x \equiv x$

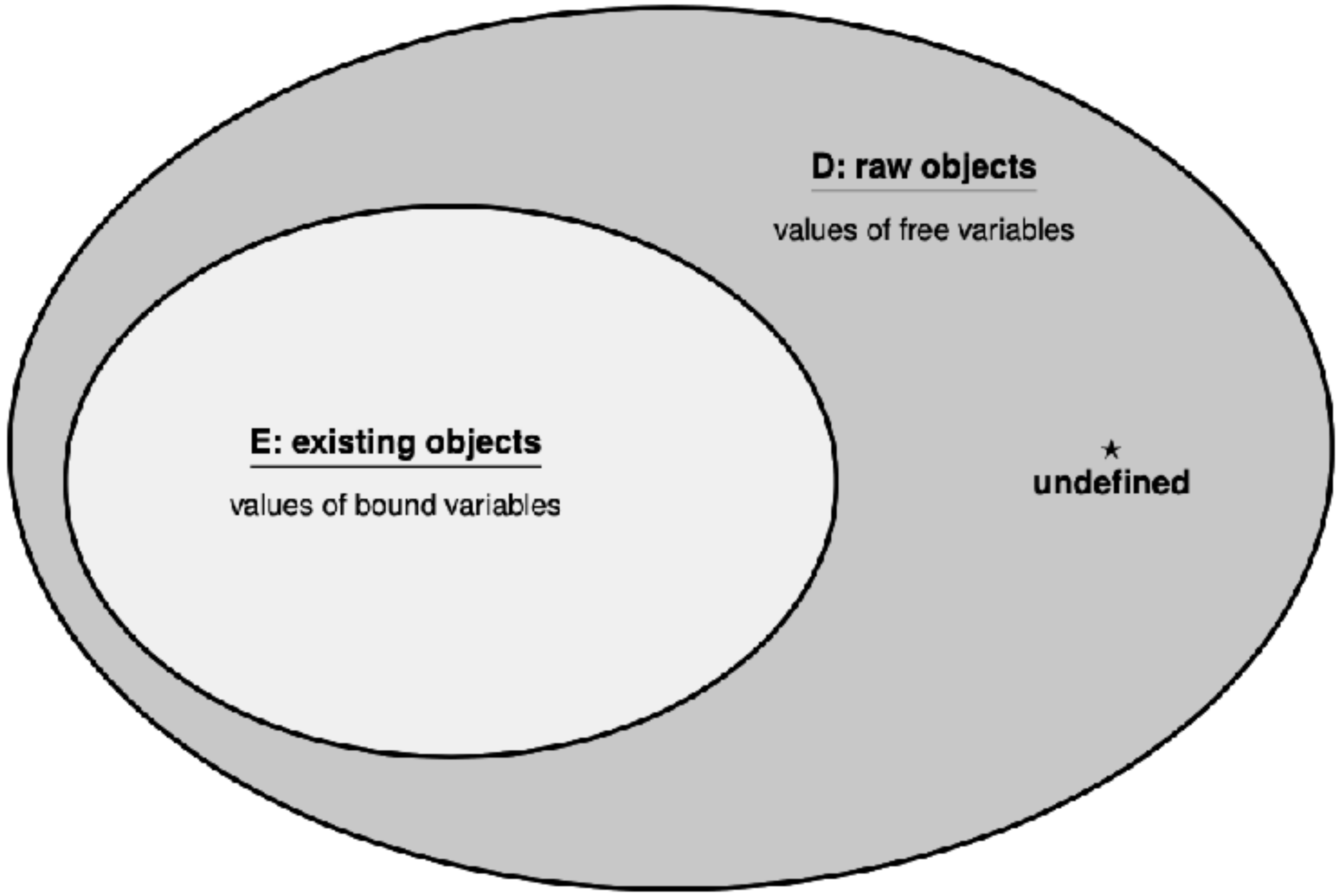
Free Logic

Existence and Description in Formal Logic (Dana Scott), 1967

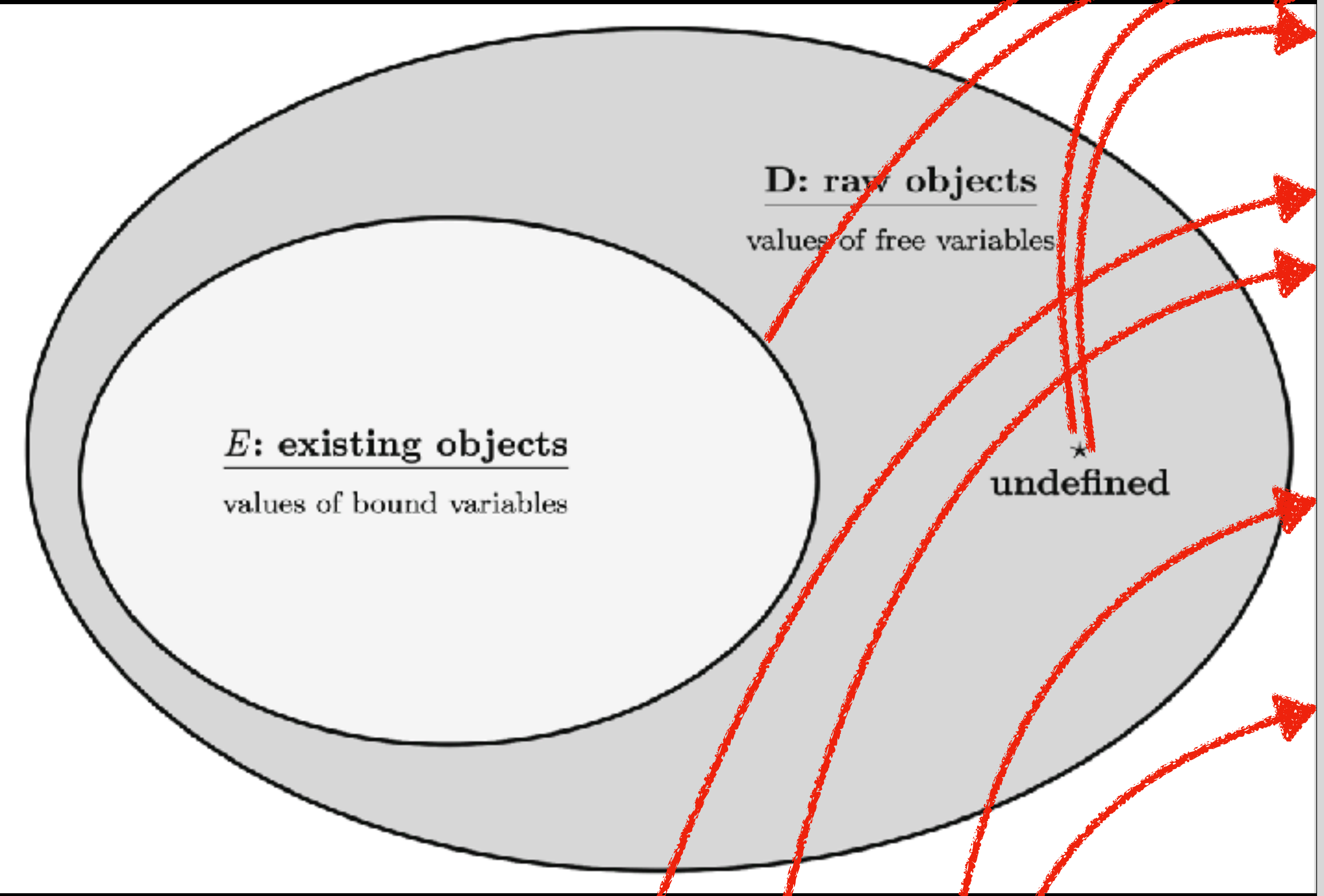
Principle 1: Bound individual variables range over domain $E \subset D$

Principle 2: Values of terms and free variables are in D, not necessarily in E only.

Principle 3: Domain E may be empty



Free Logic in HOL

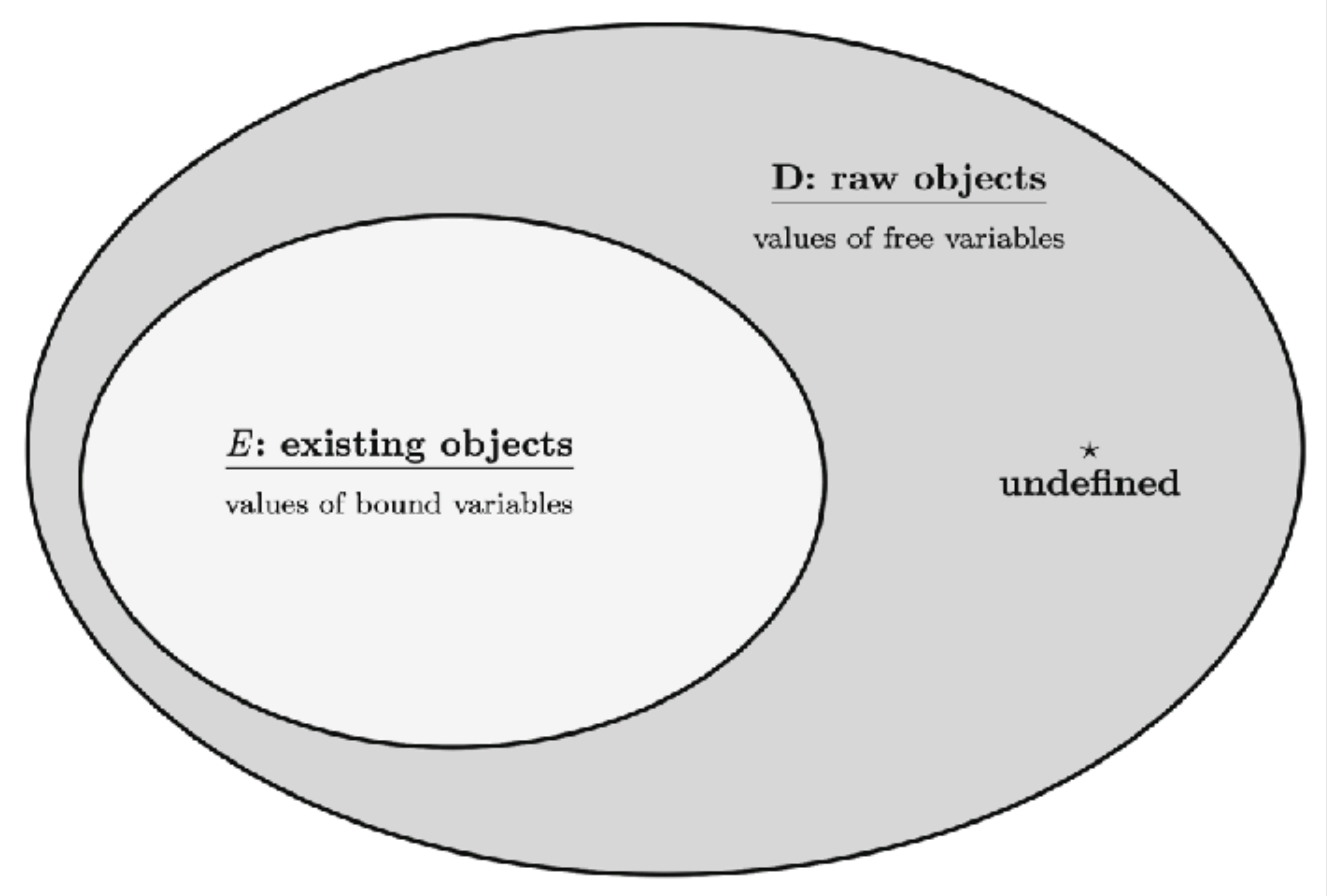


- Free connectives ¬ and → … as in HOL
- Free quantifier ∀ relativized by E
- Free description constrained by E

```
theory AxiomaticCategoryTheory_FullFreeLogic imports Main
begin
typedecl i  (*Type for individuals*)
consts fExistence:: "i⇒bool" ("E") (*Existence/definedness predicate in free logic*)
consts fStar:: "i" ("⋆")   (*Distinguished symbol for undefinedness*)
axiomatization where fStarAxiom: "¬E(⋆)" (*⋆ is a ``non-existing'' object in D.*)

abbreviation fNot ("¬") (*Free negation*)
 where "¬φ ≡ ¬φ"
abbreviation fImplies (infixr "→" 13) (*Free implication*)
 where "φ → ψ ≡ φ ⟶ ψ"
abbreviation fIdentity (infixr "=" 13) (*Free identity*)
 where "l = r ≡ l = r"
abbreviation fForall ("∀") (*Free universal quantification guarded by @{text "E"}*)
 where "∀Φ ≡ ∀x. E x ⟶ Φ x"
abbreviation fForallBinder (binder "∀" [8] 9) (*Binder notation*)
 where "∀x. φ x ≡ ∀φ"
abbreviation fThat:: "(i⇒bool)⇒i" ("I")
 where "IΦ ≡ if ∃x. E(x) ∧ Φ(x) ∧ (∀y. (E(y) ∧ Φ(y)) ⟶ (y = x))
              then THE x. E(x) ∧ Φ(x)
              else ⋆"
abbreviation fThatBinder:: "(i⇒bool)⇒i"  (binder "I" [8] 9)
 where "Ix. φ(x) ≡ I(φ)"

text ‹ Further free logic connectives can now be defined as usual. ›

abbreviation fOr (infixr "∨" 11)
 where "φ ∨ ψ ≡ (¬φ) → ψ"
abbreviation fAnd (infixr "∧" 12)
 where "φ ∧ ψ ≡ ¬(¬φ ∨ ¬ψ)"
abbreviation fImplied (infixr "←" 13)
 where "φ ← ψ ≡ ψ → φ"
abbreviation fEquiv (infixr "↔" 15)
 where "φ ↔ ψ ≡ (φ → ψ) ∧ (ψ → φ)"
abbreviation fExists ("∃")
 where "∃Φ ≡ ¬(∀(λy. ¬(Φ y)))"
abbreviation fExistsBinder (binder "∃" [8]9)
 where "∃x. φ x ≡ ∃φ"
```

As usual

Free Logic in HOL

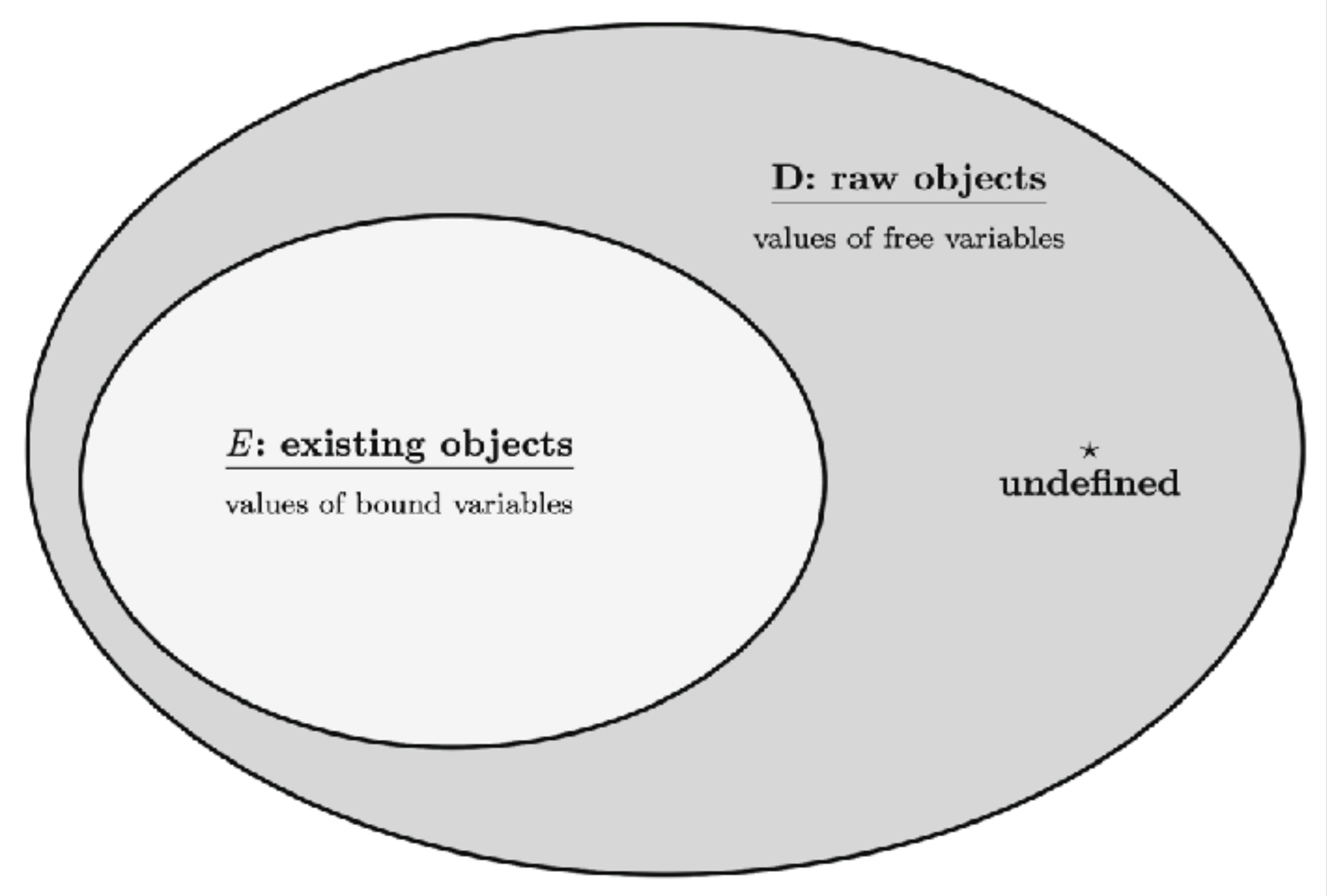



```
theory AxiomaticCategoryTheory_FullFreeLogic imports Main
begin
typedecl i  (*Type for individuals*)
consts fExistence:: "i⇒bool" ("E") (*Existence/definedness predicate in free logic*)
consts fStar:: "i" ("⋆")   (*Distinguished symbol for undefinedness*)
axiomatization where fStarAxiom: "¬E(⋆)" (*⋆ is a ``non-existing'' object in D.*)

abbreviation fNot ("¬") (*Free negation*)
 where "¬φ ≡ ¬φ"
abbreviation fImplies (infixr "→" 13) (*Free implication*)
 where "φ → ψ ≡ φ ⟶ ψ"
abbreviation fIdentity (infixr "=" 13) (*Free identity*)
 where "l = r ≡ l = r"
abbreviation fForall ("∀") (*Free universal quantification guarded by @{text "E"}*)
 where "∀Φ ≡ ∀x. E x ⟶ Φ x"
abbreviation fForallBinder (binder "∀" [8] 9) (*Binder notation*)
 where "∀x. φ x ≡ ∀φ"
abbreviation fThat:: "(i⇒bool)⇒i" ("I")
 where "IΦ ≡ if ∃x. E(x) ∧ Φ(x) ∧ (∀y. (E(y) ∧ Φ(y)) ⟶ (y = x))
             then THE x. E(x) ∧ Φ(x)
             else ⋆"
abbreviation fThatBinder:: "(i⇒bool)⇒i"  (binder "I" [8] 9)
```


```
abbreviation fForall ("∀") (*Free universal quantification guarded by @{text "E"}*)
 where "∀Φ ≡ ∀x. E x ⟶ Φ x"
abbreviation fForallBinder (binder "∀" [8] 9) (*Binder notation*)
 where "∀x. φ x ≡ ∀φ"
abbreviation fThat:: "(i⇒bool)⇒i" ("I")
 where "IΦ ≡ if ∃x. E(x) ∧ Φ(x) ∧ (∀y. (E(y) ∧ Φ(y)) ⟶ (y = x))
             then THE x. E(x) ∧ Φ(x)
             else ⋆"
abbreviation fThatBinder:: "(i⇒bool)⇒i"  (binder "I" [8] 9)
```


```
 where "Ix. φ(x) ≡ I(φ)"
```

Free Logic in HOL

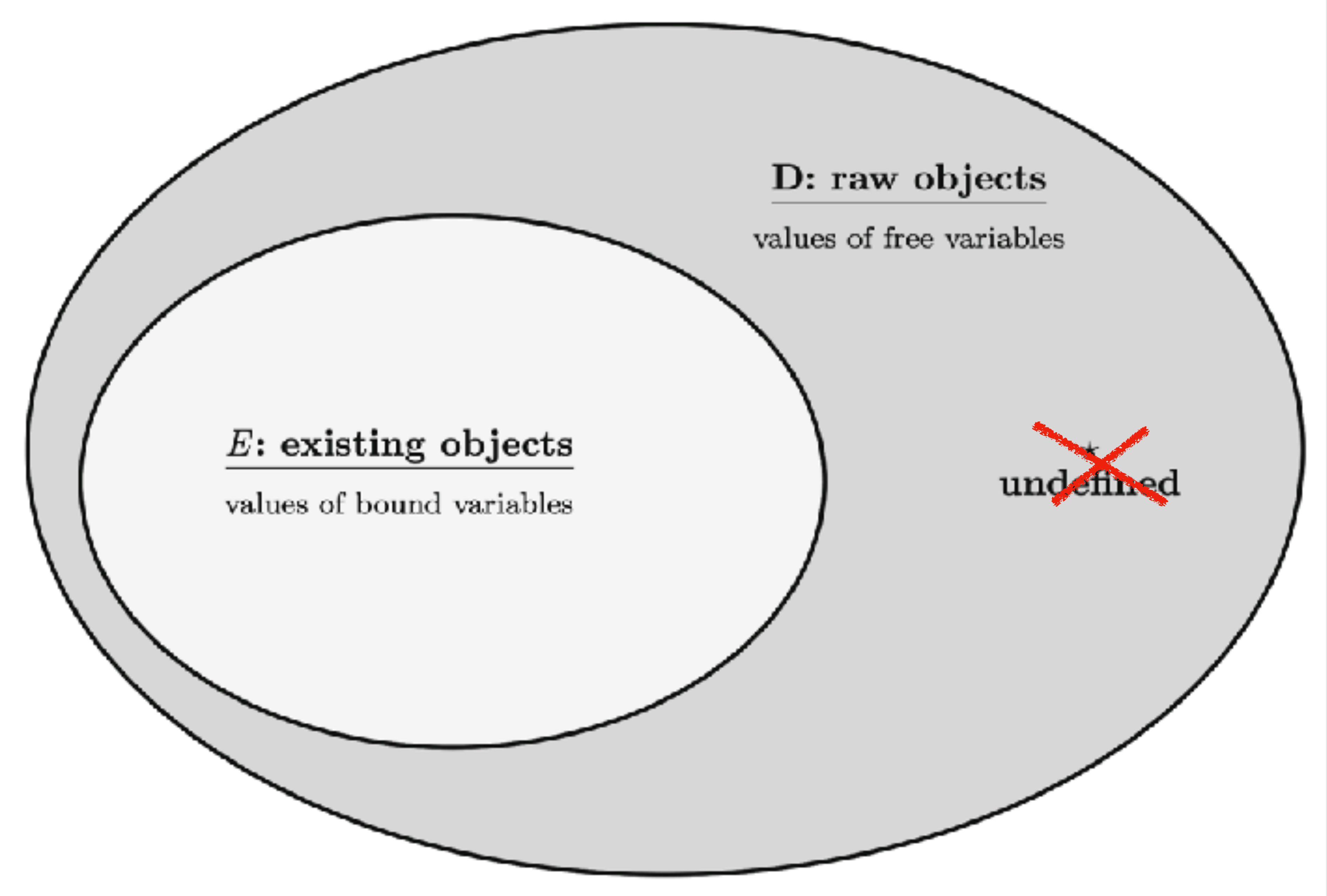


- Free connectives ¬ and → … as in HOL
- Free quantifier ∀ relativized by E
- Free description constrained by E

```
theory AxiomaticCategoryTheory_FullFreeLogic imports Main
begin
typedecl i  (*Type for individuals*)
consts fExistence:: "i⇒bool" ("E") (*Existence/definedness predicate in free logic*)
consts fStar:: "i" ("⋆")   (*Distinguished symbol for undefinedness*)
axiomatization where fStarAxiom: "¬E(⋆)" (*⋆ is a ``non-existing'' object in D.*)

abbreviation fNot ("¬") (*Free negation*)
 where "¬φ ≡ ¬φ"
abbreviation fImplies (infixr "→" 13) (*Free implication*)
 where "φ → ψ ≡ φ ⟶ ψ"
abbreviation fIdentity (infixr "=" 13) (*Free identity*)
 where "l = r ≡ l = r"
abbreviation fForall ("∀") (*Free universal quantification guarded by @{text "E"}*)
 where "∀Φ ≡ ∀x. E x ⟶ Φ x"
abbreviation fForallBinder (binder "∀" [8] 9) (*Binder notation*)
 where "∀x. φ x ≡ ∀φ"
abbreviation fThat:: "(i⇒bool)⇒i" ("I")
 where "IΦ ≡ if ∃x. E(x) ∧ Φ(x) ∧ (∀y. (E(y) ∧ Φ(y)) ⟶ (y = x))
             then THE x. E(x) ∧ Φ(x)
             else ⋆"
abbreviation fThatBinder:: "(i⇒bool)⇒i"  (binder "I" [8] 9)
 where "Ix. φ(x) ≡ I(φ)"

text ‹ Further free logic connectives can now be defined as usual. ›

abbreviation fOr (infixr "∨" 11)
 where "φ ∨ ψ ≡ (¬φ) → ψ"
abbreviation fAnd (infixr "∧" 12)
 where "φ ∧ ψ ≡ ¬(¬φ ∨ ¬ψ)"
abbreviation fImplied (infixr "←" 13)
 where "φ ← ψ ≡ ψ → φ"
abbreviation fEquiv (infixr "↔" 15)
 where "φ ↔ ψ ≡ (φ → ψ) ∧ (ψ → φ)"
abbreviation fExists ("∃")
 where "∃Φ ≡ ¬(∀(λy. ¬(Φ y)))"
abbreviation fExistsBinder (binder "∃" [8]9)
 where "∃x. φ x ≡ ∃φ"
```

As usual

Free Logic in HOL



- Free connectives ¬ and → … as in HOL
- Free quantifier ∀ relativized by E
- Free description constrained by E

```
theory AxiomaticCategoryTheory_FullFreeLogic imports Main
begin
typedecl i  (*Type for individuals*)
consts fExistence:: "i⇒bool" ("E") (*Existence/definedness predicate in free logic*)
consts fStar:: "i" ("⋆")   (*Distinguished symbol for undefinedness*)
axiomatization where fStarAxiom: "¬E(⋆)" (*⋆ is a `non-existing' object in D.*)

abbreviation fNot ("¬") (*Free negation*)
 where "¬φ ≡ ¬φ"
abbreviation fImplies (infixr "→" 13) (*Free implication*)
 where "φ → ψ ≡ φ ⟶ ψ"
abbreviation fIdentity (infixr "=" 13) (*Free identity*)
 where "l = r ≡ l = r"
abbreviation fForall ("∀") (*Free universal quantification guarded by @{text "E"}*)
 where "∀Φ ≡ ∀x. E x ⟶ Φ x"
abbreviation fForallBinder (binder "∀" [8] 9) (*Binder notation*)
 where "∀x. φ x ≡ ∀φ"
abbreviation fThat:: "(i⇒bool)⇒i" ("I")
 where "IΦ ≡ if ∃x. E(x) ∧ Φ(x) ∧ (∀y. (E(y) ∧ Φ(y)) ⟶ (y = x))
               then THE x. E(x) ∧ Φ(x)
               else ⋆"
abbreviation fThatBinder:: "(i⇒bool)⇒i"  (binder "I" [8] 9)
 where "Ix. φ(x) ≡ I(φ)"

text ‹ Further free logic connectives can now be defined as usual. ›

abbreviation fOr (infixr "∨" 11)
 where "φ ∨ ψ ≡ (¬φ) → ψ"
abbreviation fAnd (infixr "∧" 12)
 where "φ ∧ ψ ≡ ¬(¬φ ∨ ¬ψ)"
abbreviation fImplied (infixr "←" 13)
 where "φ ← ψ ≡ ψ → φ"
abbreviation fEquiv (infixr "↔" 15)
 where "φ ↔ ψ ≡ (φ → ψ) ∧ (ψ → φ)"
abbreviation fExists ("∃")
 where "∃Φ ≡ ¬(∀(λy. ¬(Φ y)))"
abbreviation fExistsBinder (binder "∃" [8]9)
 where "∃x. φ x ≡ ∃φ"
```

As usual

Case Study in
Axiomatic Category Theory

Axioms Set I

Generalized Monoids

Dana Scott

Axioms Set II

Axioms Set I

Generalized Monoids

Dana Scott

Axioms Set I

Generalized Monoids

Axioms Set II

Axioms Set III

Dana Scott

Axioms Set I

Generalized Monoids

Axioms Set II

Axioms Set III

Axioms Set IV

Dana Scott

Axioms Set I

Generalized Monoids

Axioms Set II

Axioms Set III

Axioms Set IV

Axioms Set V

Dana Scott's Axioms from 1977

Dana Scott

Axioms Set I

Generalized Monoids

Axioms Set II

Axioms Set III

Axioms Set IV

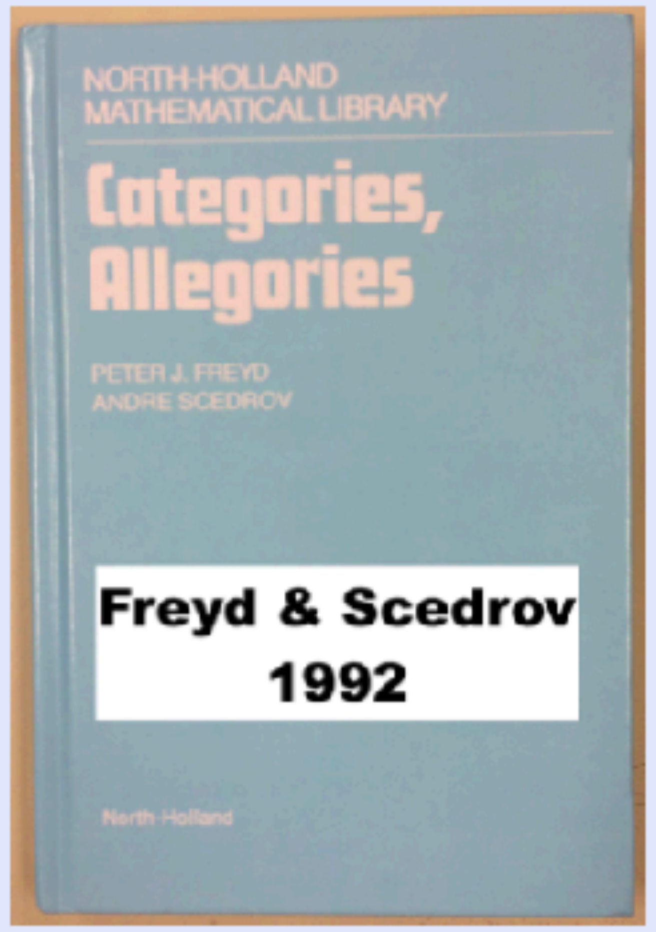
Axioms Set V

Dana Scott's Axioms from 1977

—?—

Axioms Set VI

Freyd & Scedrov's Axioms from 1992



Dana Scott

Axioms Set I

Generalized Monoids

Axioms Set II

Axioms Set III

Axioms Set IV

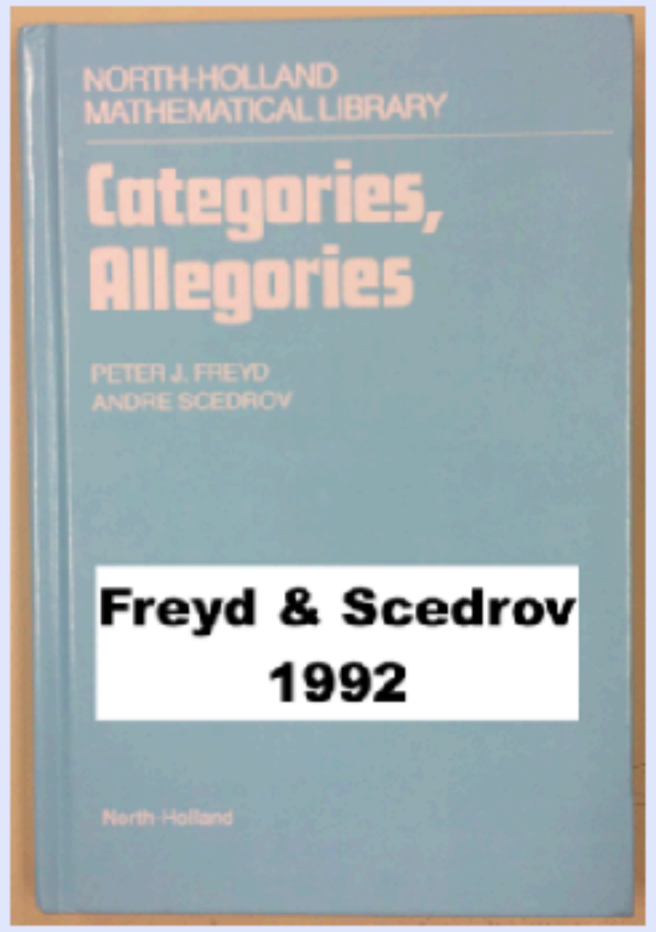
Axioms Set V

Dana Scott's Axioms from 1977

—?—

Axioms Set VI

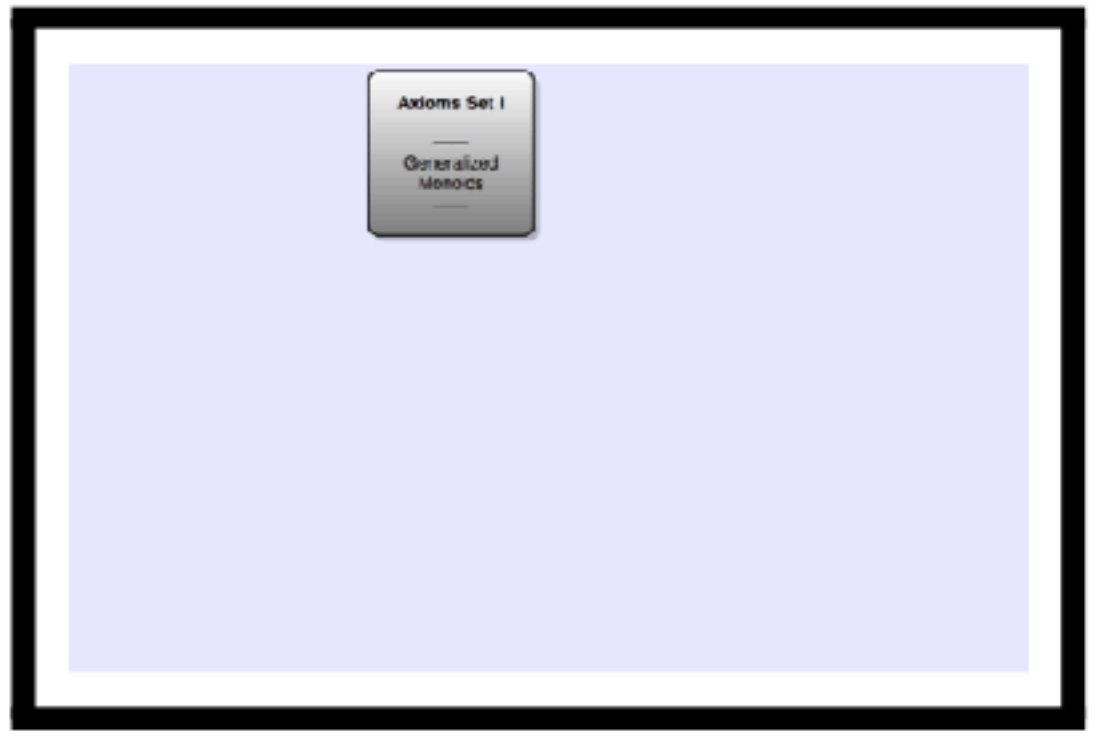
Freyd & Scedrov's Axioms from 1992



Dana Scott

all equivalent?

Preliminaries



Morphisms: objects of type of i (raw domain D)

Partial functions:

domain	dom	of type $i \rightarrow i$
codomain	cod	of type $i \rightarrow i$
composition	$\cdot$	of type $i \rightarrow i \rightarrow i$ (resp. $i \times i \rightarrow i$)

$\cong$ denotes Kleene equality: $x \cong y \equiv (Ex \vee Ey) \rightarrow x = y$

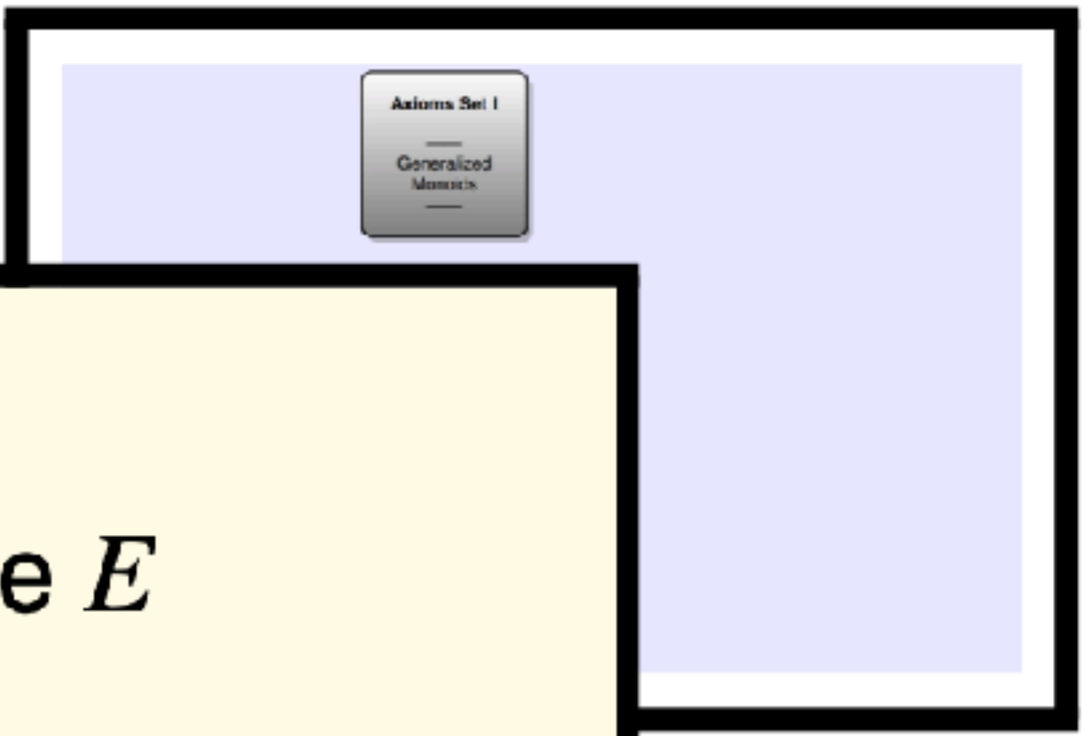
(where $=$ is identity on all objects of type i, existing or non-existing)

$\cong$ is an equivalence relation: **Sledgehammer.**

$\simeq$ denotes existing identity: $x \simeq y \equiv Ex \wedge Ey \wedge x = y$

$\simeq$ is symmetric and transitive, but lacks reflexivity: **Sledgehammer**, **Nitpick**.

Preliminaries



- $\simeq$ equivalence relation on E, empty relation outside E
- $1/0 \not\simeq 1/0 \quad 1/0 \not\simeq 2/0 \quad \ldots$
- $Ix.pkoFrance(x) \not\simeq Ix.pkoFrance(x)$
  $Ix.pkoFrance(x) \not\simeq Ix.pkoPoland(x)$

$\cong$ denotes Kleene equality: $\quad x \cong y \equiv (Ex \vee Ey) \rightarrow x = y$

(where $=$ is identity on all objects of type i, existing or non-existing)

$\cong$ is an equivalence relation: **Sledgehammer.**

$\simeq$ denotes existing identity: $\quad x \simeq y \equiv Ex \wedge Ey \wedge x = y$

$\simeq$ is symmetric and transitive, but lacks reflexivity: **Sledgehammer**, **Nitpick**.

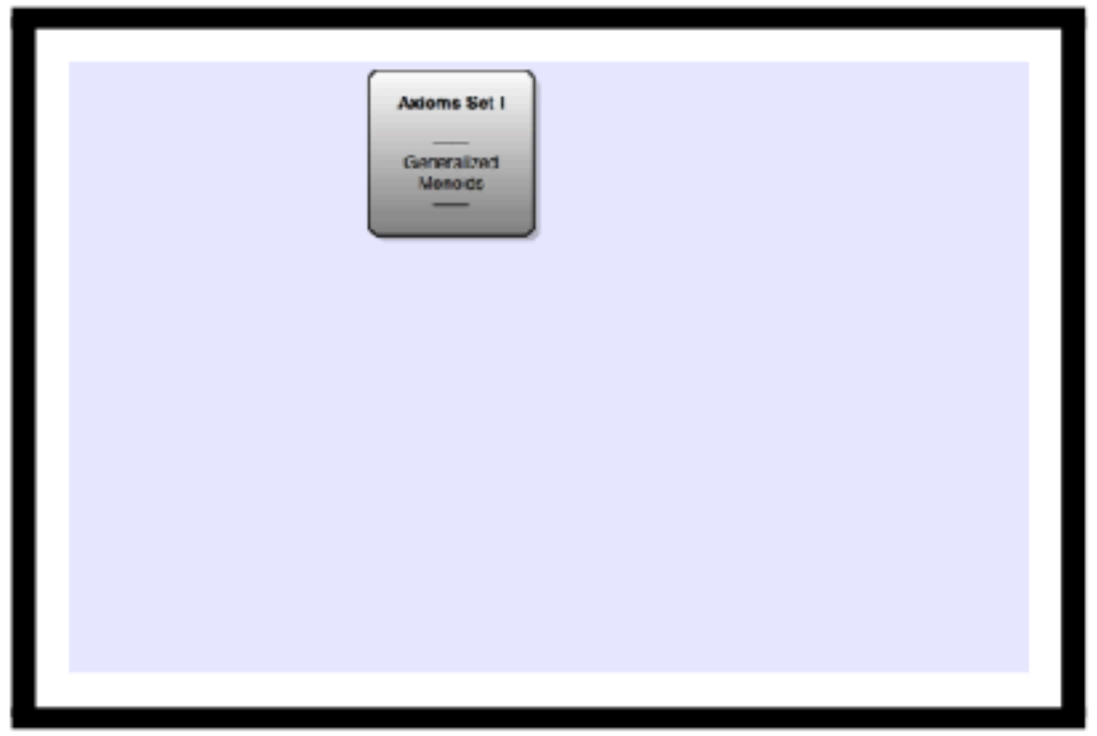
Category Theory in Free Logic (in HOL)

```
section ‹ Axioms Set V ›

locale Axioms_Set_V =
assumes
  S1: "E(dom x) ⟶ E x" and
  S2: "E(cod y) ⟶ E y" and
  S3: "E(x·y) ↔ dom x ≃ cod y" and
  S4: "x·(y·z) ≅ (x·y)·z" and
  S5: "x·(dom x) ≅ x" and
  S6: "(cod y)·y ≅ y"
begin  (*The obligatory consistency checks*)
  lemma True
     nitpick [satisfy, user_axioms, expect=genuine] oops (*model found*)
  lemma assumes "∃x. ¬(E x)" shows True
    nitpick [satisfy, user_axioms, expect=genuine] oops (*model found*)
  lemma assumes "(∃x. ¬(E x)) ∧ (∃x. (E x))" shows True
    nitpick [satisfy, user_axioms,  expect=genuine] oops (*model found*)
end

context Axioms_Set_V   (*Axioms Set IV is implied by Axioms Set V*)
begin
  lemma S_ivFromV: "(E(x·y) ⟶ (E x ∧ E y)) ∧ (E(dom x ) ⟶ E x) ∧ (E(cod y) ⟶ E y)"
    using S1 S2 S3 by blast
  lemma E_ivFromV: "E(x·y) ↔ (dom x ≅ cod y ∧ E(cod y))"  using S3 by metis
  lemma A_ivFromV: "x·(y·z) ≅ (x·y)·z"  using S4 by blast
  lemma C_ivFromV: "(cod y)·y ≅ y"  using S6 by blast
  lemma D_ivFromV: "x·(dom x) ≅ x"  using S5 by blast
end

context Axioms_Set_IV  (*Axioms Set V is implied by Axioms Set IV*)
begin
  lemma S1FromIV:  "E(dom x) ⟶ E x"  using S_iv by blast
  lemma S2FromIV:  "E(cod y) ⟶ E y"  using S_iv by blast
  lemma S3FromIV:  "E(x·y) ↔ dom x ≃ cod y"  using E_iv by metis
  lemma S4FromIV:  "x·(y·z) ≅ (x·y)·z"  using A_iv by blast
  lemma S5FromIV:  "x·(dom x) ≅ x"  using D_iv by blast
  lemma S6FromIV:  "(cod y)·y ≅ y"  using C_iv by blast
```

Table 1 Stepwise evolution of Scott's [33] axiom system for category theory from partial monoids

Axioms Set I	
S_i	$E(x\cdot y) \longrightarrow (Ex \wedge Ey)$
E_i	$E(x\cdot y) \longleftarrow (Ex \wedge Ey \wedge (\exists z. z\cdot z \cong z \wedge x\cdot z \cong x \wedge z\cdot y \cong y))$
A_i	$x\cdot(y\cdot z) \cong (x\cdot y)\cdot z$
C_i	$\forall y. \exists i. Ii \wedge i\cdot y \cong y$
D_i	$\forall x. \exists j. Ij \wedge x\cdot j \cong x$
Axioms Set II	
S_{ii}	$E(x\cdot y) \longrightarrow (Ex \wedge Ey) \wedge (E(dom\ x) \longrightarrow Ex) \wedge (E(cod\ y) \longrightarrow Ey)$
E_{ii}	$E(x\cdot y) \longleftarrow (Ex \wedge Ey \wedge (\exists z. z\cdot z \cong z \wedge x\cdot z \cong x \wedge z\cdot y \cong y))$
A_{ii}	$x\cdot(y\cdot z) \cong (x\cdot y)\cdot z$
C_{ii}	$Ey \longrightarrow (I(cod\ y) \wedge (cod\ y)\cdot y \cong y)$
D_{ii}	$Ex \longrightarrow (I(dom\ x) \wedge x\cdot(dom\ x) \cong x)$
Axioms Set III	
S_{iii}	$E(x\cdot y) \longrightarrow (Ex \wedge Ey) \wedge (E(dom\ x) \longrightarrow Ex) \wedge (E(cod\ y) \longrightarrow Ey)$
E_{iii}	$E(x\cdot y) \longleftarrow (dom\ x \cong cod\ y \wedge E(cod\ y)))$
A_{iii}	$x\cdot(y\cdot z) \cong (x\cdot y)\cdot z$
C_{iii}	$Ey \longrightarrow (I(cod\ y) \wedge (cod\ y)\cdot y \cong y)$
D_{iii}	$Ex \longrightarrow (I(dom\ x) \wedge x\cdot(dom\ x) \cong x)$
Axioms Set IV	
S_{iv}	$E(x\cdot y) \longrightarrow (Ex \wedge Ey) \wedge (E(dom\ x) \longrightarrow Ex) \wedge (E(cod\ y) \longrightarrow Ey)$
E_{iv}	$E(x\cdot y) \longleftrightarrow (dom\ x \cong cod\ y \wedge E(cod\ y)))$
A_{iv}	$x\cdot(y\cdot z) \cong (x\cdot y)\cdot z$
C_{iv}	$(cod\ y)\cdot y \cong y$
D_{iv}	$x\cdot(dom\ x) \cong x$
Axioms Set V [33]	
$S1$	$E(dom\ x) \longrightarrow Ex$
$S2$	$E(cod\ y) \longrightarrow Ey$
$S3$	$E(x\cdot y) \longleftrightarrow dom\ x \simeq cod\ y$
$S4$	$x\cdot(y\cdot z) \cong (x\cdot y)\cdot z$
$S5$	$(cod\ y)\cdot y \cong y$
$S6$	$x\cdot(dom\ x) \cong x$

From Monoids to Categories



We employ a partial, strict binary composition operation $\cdot$
Left and right identity elements are addressed in C_i, D_i, .

Categories: Axioms Set I

S_i	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey)$
E_i	Existence	$E(x \cdot y) \leftarrow (Ex \wedge Ey \wedge (\exists z. z \cdot z \cong z \wedge x \cdot z \cong x \wedge z \cdot y \cong y))$
A_i	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_i	Codomain	$\forall y. \exists i. ID(i) \wedge i \cdot y \cong y$
D_i	Domain	$\forall x. \exists j. ID(j) \wedge x \cdot j \cong x$

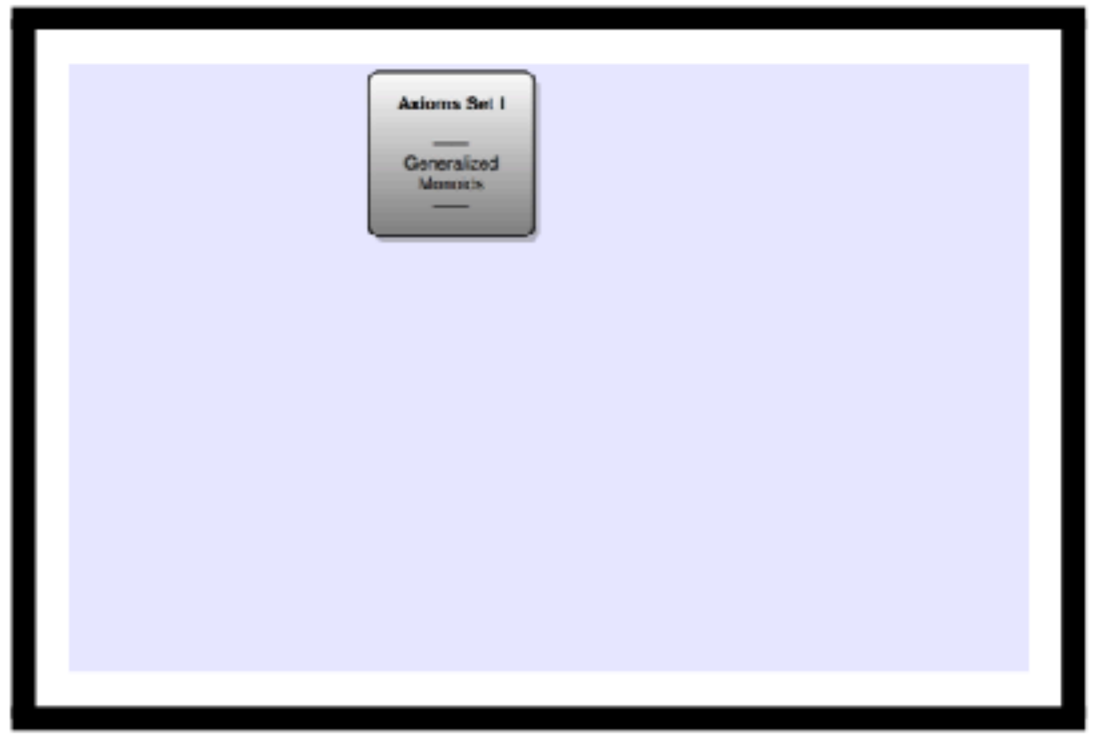
where I is an identity morphism predicate:

$$ID(i) \equiv (\forall x.\ E(i \cdot x) \rightarrow i \cdot x \cong x) \wedge (\forall x.\ E(x \cdot i) \rightarrow x \cdot i \cong x)$$

Monoid

Closure:	$\forall a, b \in S.\ a \circ b \in S$
Associativity:	$\forall a, b, c \in S.\ a \circ (b \circ c) = (a \circ b) \circ c$
Identity:	$\exists id_S \in S.\ \forall a \in S.\ id_S \circ a = a = a \circ id_S$

From Monoids to Categories



We employ a partial, strict binary composition operation $\cdot$
Left and right identity elements are addressed in C_i, D_i, .

Categories: Axioms Set I

S_i	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey)$
E_i	Existence	$E(x \cdot y) \leftarrow (Ex \wedge Ey \wedge (\exists z. z \cdot z \cong z \wedge x \cdot z \cong x \wedge z \cdot y \cong y))$
A_i	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_i	Codomain	$\forall y. \exists i. ID(i) \wedge i \cdot y \cong y$
D_i	Domain	$\forall x. \exists j. ID(j) \wedge x \cdot j \cong x$

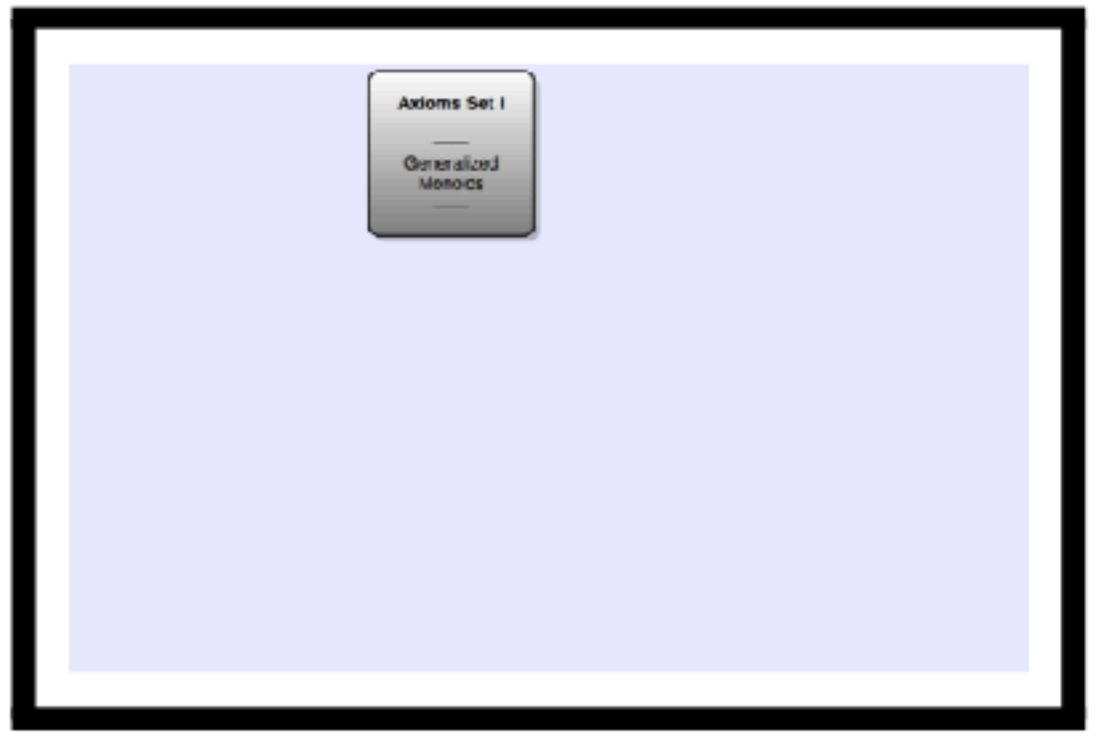
where I is an identity morphism predicate:

$$ID(i) \equiv (\forall x.\ E(i \cdot x) \rightarrow i \cdot x \cong x) \wedge (\forall x.\ E(x \cdot i) \rightarrow x \cdot i \cong x)$$

Experiments with Isabelle/HOL

- The i in axiom C is unique: **Sledgehammer**.
- The j in axiom D is unique: **Sledgehammer**.
- However, the i and j need not be equal: **Nitpick**

From Monoids to Categories



We employ a partial, strict binary composition operation $\cdot$
Left and right identity elements are addressed in C_i, D_i, .

Categories: Axioms Set I

S_i	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey)$
E_i	Existence	$E(x \cdot y) \leftarrow (Ex \wedge Ey \wedge (\exists z. z \cdot z \cong z \wedge x \cdot z \cong x \wedge z \cdot y \cong y))$
A_i	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_i	Codomain	$\forall y. \exists i. ID(i) \wedge i \cdot y \cong y$
D_i	Domain	$\forall x. \exists j. ID(j) \wedge x \cdot j \cong x$

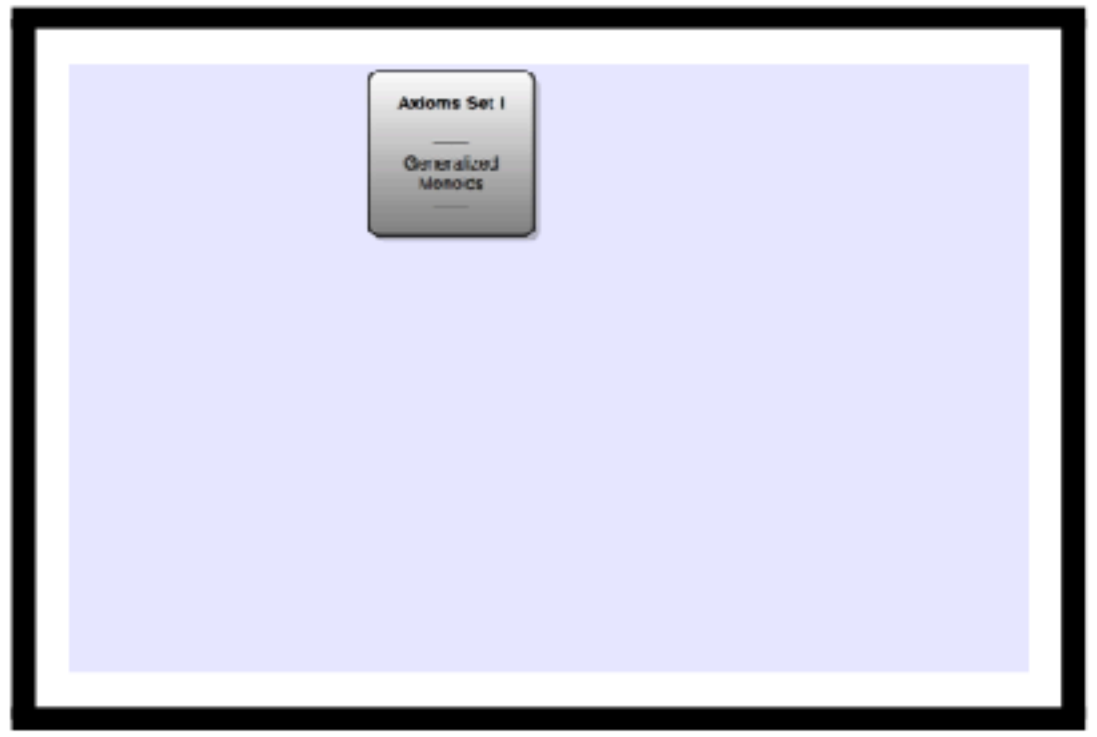
where I is an identity morphism predicate:

$$ID(i) \equiv (\forall x.\ E(i \cdot x) \rightarrow i \cdot x \cong x) \wedge (\forall x.\ E(x \cdot i) \rightarrow x \cdot i \cong x)$$

Experiments with Isabelle/HOL

- The left-to-right direction of E is implied: **Sledgehammer**.
  $$E(x \cdot y) \rightarrow (Ex \wedge Ey \wedge (\exists z. z \cdot z \cong z \wedge x \cdot z \cong x \wedge z \cdot y \cong y))$$

From Monoids to Categories



We employ a partial, strict binary composition operation $\cdot$
Left and right identity elements are addressed in C_i, D_i, .

Categories: Axioms Set I

S_i	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey)$
E_i	Existence	$E(x \cdot y) \leftarrow (Ex \wedge Ey \wedge (\exists z. z \cdot z \cong z \wedge x \cdot z \cong x \wedge z \cdot y \cong y))$
A_i	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_i	Codomain	$\forall y. \exists i. ID(i) \wedge i \cdot y \cong y$
D_i	Domain	$\forall x. \exists j. ID(j) \wedge x \cdot j \cong x$

where I is an identity morphism predicate:

$$ID(i) \equiv (\forall x.\ E(i \cdot x) \rightarrow i \cdot x \cong x) \wedge (\forall x.\ E(x \cdot i) \rightarrow x \cdot i \cong x)$$

Experiments with Isabelle/HOL

- Model finder **Nitpick** confirms that this axiom set is consistent.
- Even if we assume there are non-existing objects $(\exists x. \neg(Ex))$ we get consistency.

Interaction: Dana – Christoph – Isabelle/HOL

Dana Scott <dana.scott@cs.cmu.edu> 8/6/16

to me

> On Aug 5, 2016, at 11:00 PM, Christoph Benzmueller
>
> When we take IDD(i) as
> (all x)[E(i.x) ==> i.x == x] &
> (all x)[E(x.i) ==> x.i == x]
> and replace ID(i) in our SACDE-axioms by IDD(i) the
> ID(I) and IDD(i) are equivalent. See attachment New_
>
> So IDD(i) seem suited as a notion of identity morphis

Ha! I am surprised, because I did not see how to prove

(all i)[IDD(i) ==> i.i == i]

I have to think about this. I hate it when computers are smarter than I am!

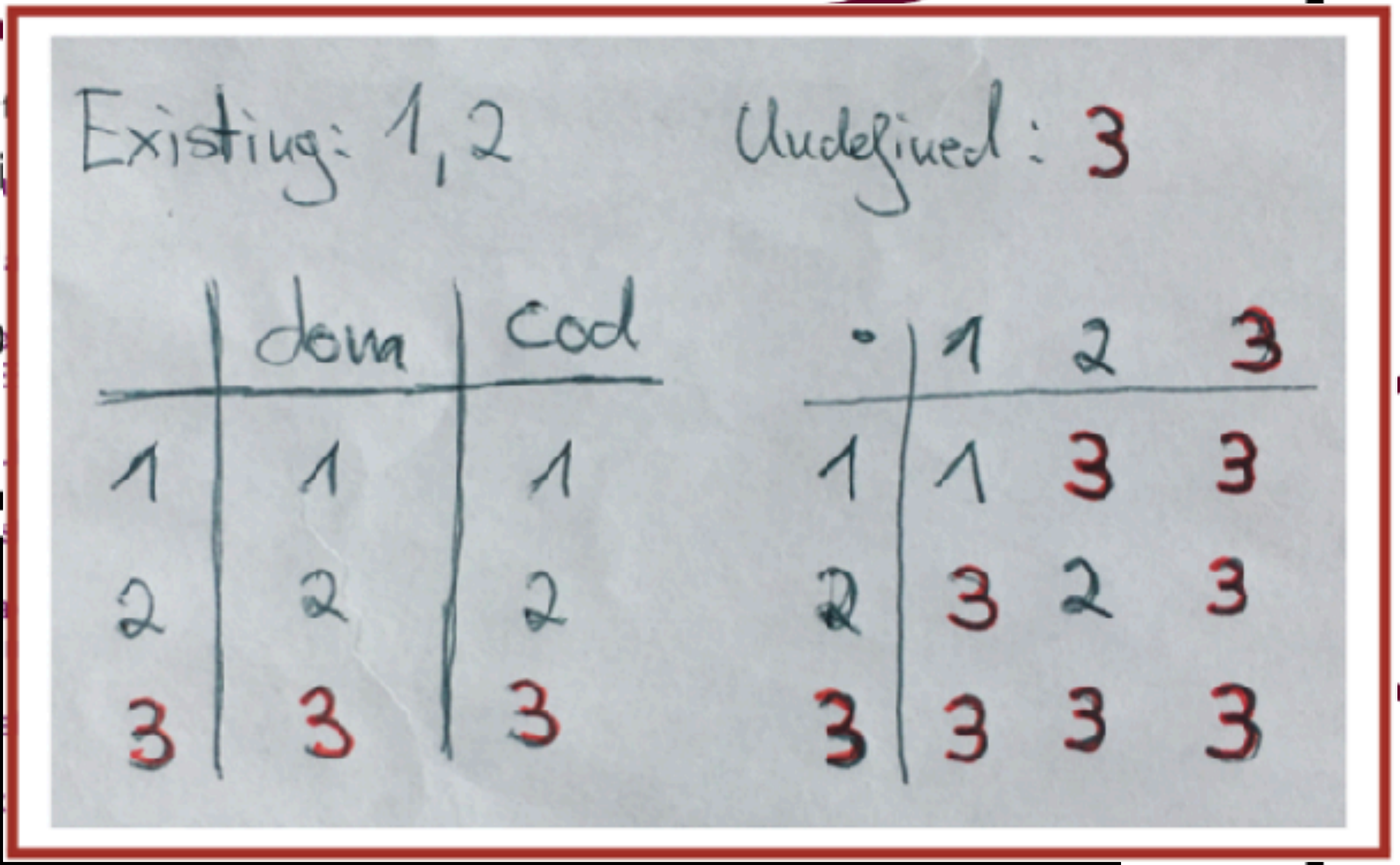
I guess C and D have to be used.

Chri

to D

Hi D

C.



Interaction: Dana – Christoph – Isabelle/HOL

Christoph Benzmueller <c.benzmueller@gmail.com> 7/23/16

to Dana

Dana,

here are the results of the experiments; doesn't look too good.

On Fri, Jul 22, 2016 at 11:43 PM, Dana Scott <dana.scott@cs.cmu.edu> wrote:

> On Jul 21, 2016, at 9:32 AM, Christoph Benzmueller <c.benzmueller@gmail.com> wrote:
>
> The F-axioms are all provable from the old S-axioms.
> But D2, D3 and E3 are not.

I think I see the trouble with those D axioms. But E3 is very odd.

E3: E(x.y) ==> (exist i)[Id(i) & x.(i.y) == x.y]

You see, by the S-axioms, if you assume E(x.y), then E(x) & E(y) & E(cod(x)) follows. So the "i" in the conclusion of E3 ought to be "cod(x)".

Please check, therefore, whether this is provable from the S-axioms:

(all x) Id(cod(x))

Apparently it isn't. See file Scott_new_axioms_4.png; the countermodel is presented in the lower window; he have:

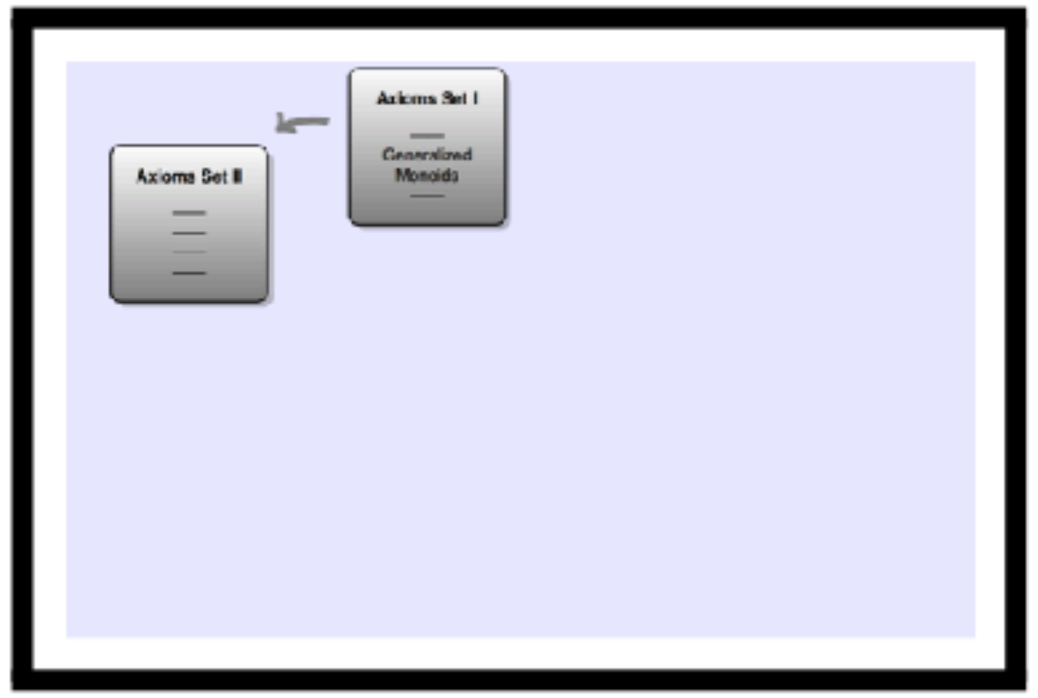
dom(i1)=i1, dom(i2)=i2, dom(i3)=i3
cod(i1)=i1, cod(i2)=i2, cod(i3)=i3
i1.i1=i1, i1.i2=i3, i1.i3=i3
i2.i1=i3, i2.i2=i2, i2.i3=i3
i3.i1=i3, i3.i2=i3, i3.i3=i3
E(i1),E(i2), ~E(i3)

Countermodel by Nitpick converted by me into a readable form

I have briefly checked it; it seems to validate each S-axiom.

If this is OK, then E3 should have been provable.

From Monoids to Categories



Axioms Set II is developed from Axioms Set I by Skolemization of i and j in axioms C and D. We can argue semantically that every model of Axioms Set I has such functions. The strictness axiom S is extended, so that strictness is now also postulated for the new Skolem functions dom and cod.

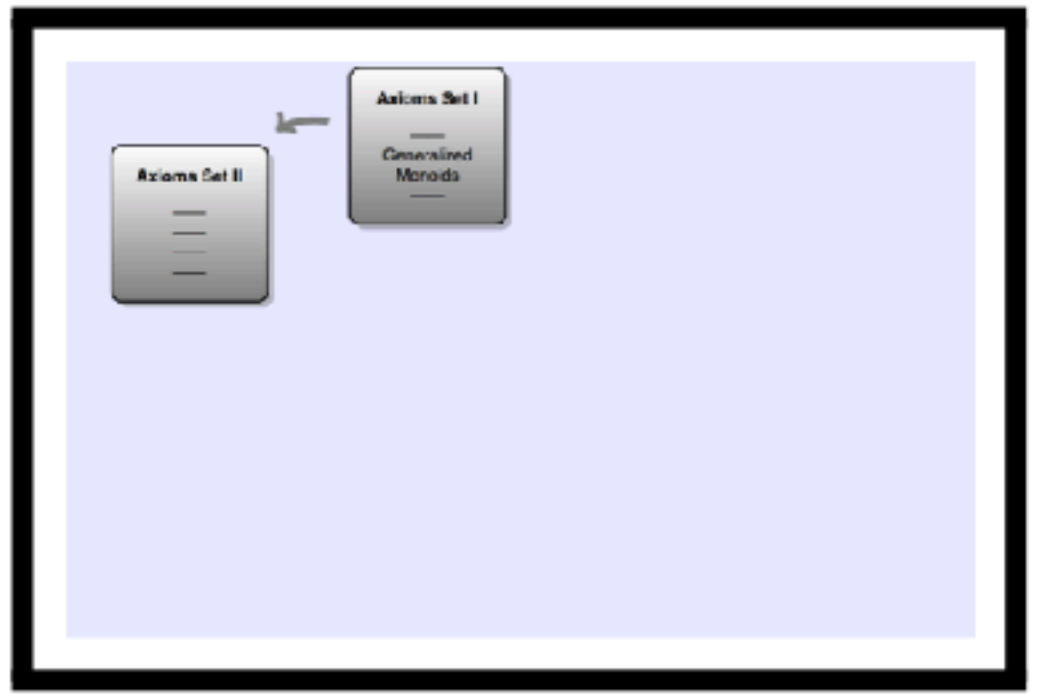
Categories: Axioms Set II

S_{ii}	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey) \wedge (E(dom\ x) \rightarrow Ex) \wedge (E(cod\ y) \rightarrow Ey)$
E_{ii}	Existence	$E(x \cdot y) \leftarrow (Ex \wedge Ey \wedge (\exists z. z \cdot z \cong z \wedge x \cdot z \cong x \wedge z \cdot y \cong y))$
A_{ii}	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_{ii}	Codomain	$Ey \rightarrow (ID(cod\ y) \wedge (cod\ y) \cdot y \cong y)$
D_{ii}	Domain	$Ex \rightarrow (ID(dom\ x) \wedge x \cdot (dom\ x) \cong x)$

Categories: Axioms Set I

S_i	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey)$
E_i	Existence	$E(x \cdot y) \leftarrow (Ex \wedge Ey \wedge (\exists z. z \cdot z \cong z \wedge x \cdot z \cong x \wedge z \cdot y \cong y))$
A_i	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_i	Codomain	$\forall y. \exists i. ID(i) \wedge i \cdot y \cong y$
D_i	Domain	$\forall x. \exists j. ID(j) \wedge x \cdot j \cong x$

From Monoids to Categories



Axioms Set II is developed from Axioms Set I by Skolemization of i and j in axioms C and D. We can argue semantically that every model of Axioms Set I has such functions. The strictness axiom S is extended, so that strictness is now also postulated for the new Skolem functions dom and cod.

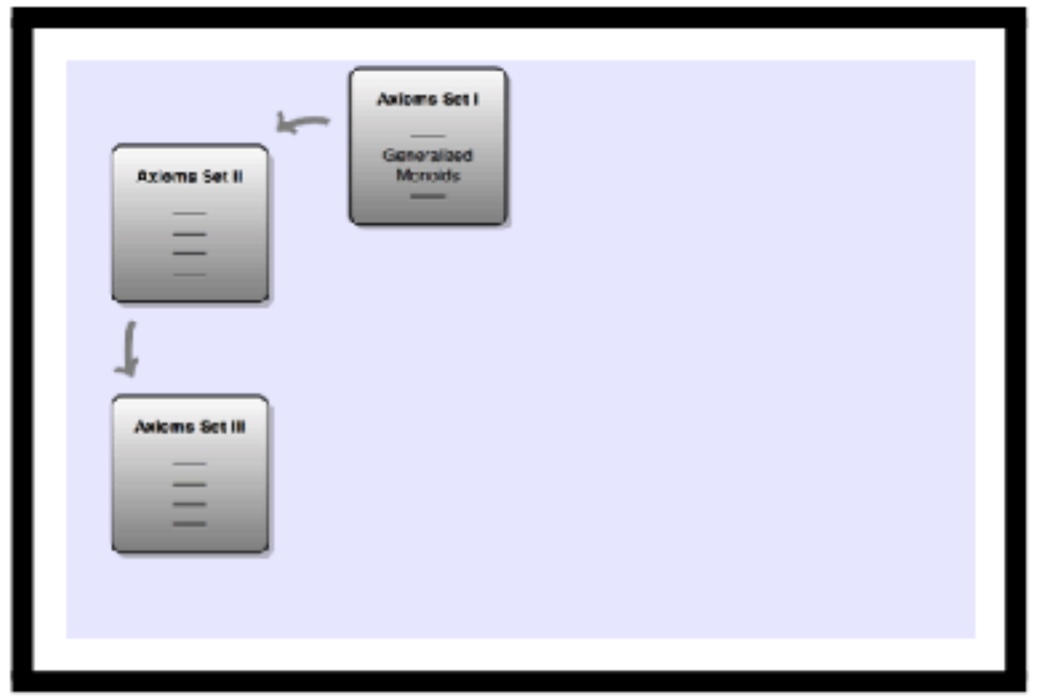
Categories: Axioms Set II

S_{ii}	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey) \wedge (E(dom\ x) \rightarrow Ex) \wedge (E(cod\ y) \rightarrow Ey)$
E_{ii}	Existence	$E(x \cdot y) \leftarrow (Ex \wedge Ey \wedge (\exists z. z \cdot z \cong z \wedge x \cdot z \cong x \wedge z \cdot y \cong y))$
A_{ii}	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_{ii}	Codomain	$Ey \rightarrow (ID(cod\ y) \wedge (cod\ y) \cdot y \cong y)$
D_{ii}	Domain	$Ex \rightarrow (ID(dom\ x) \wedge x \cdot (dom\ x) \cong x)$

Experiments with Isabelle/HOL

- Consistency holds (also when $\exists x. \neg(Ex)$): confirmed by **Nitpick**.
- Axioms Set II implies Axioms Set I: easily proved by **Sledgehammer**.
- Axioms Set I also implies Axioms Set II (by semantical means on the meta-level)

From Monoids to Categories



In Axioms Set III the existence axiom E is simplified by taking advantage of the two new Skolem functions dom and cod.

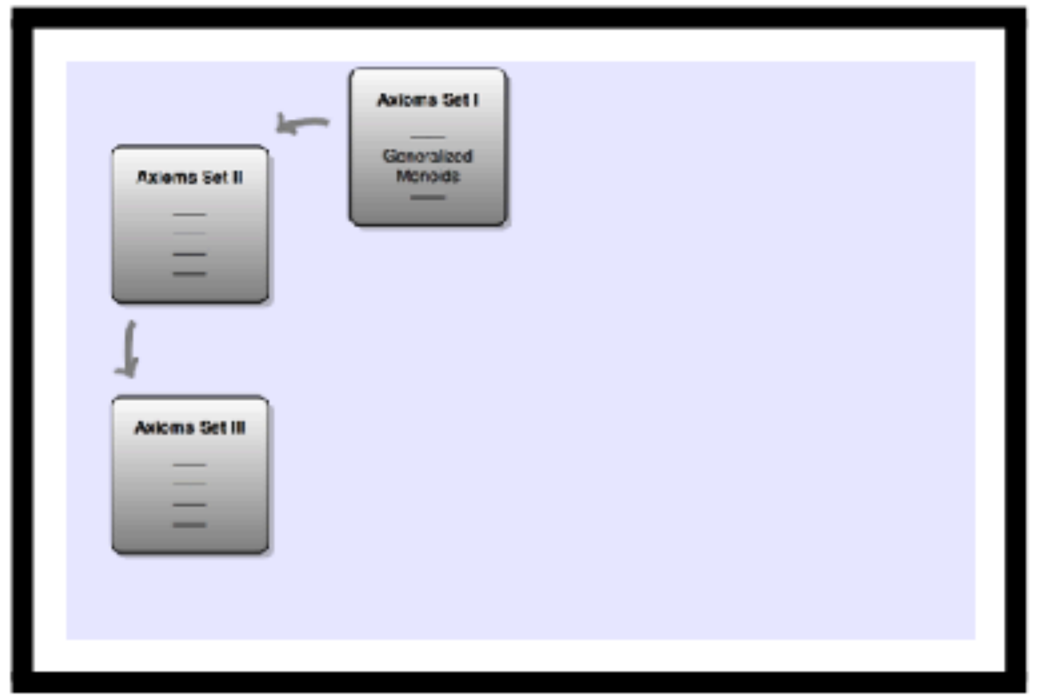
Categories: Axioms Set III

S_{iii}	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey) \wedge (E(dom\ x) \rightarrow Ex) \wedge (E(cod\ y) \rightarrow Ey)$
E_{iii}	Existence	$E(x \cdot y) \leftarrow (dom\ x \cong cod\ y \wedge E(cod\ y))$
A_{iii}	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_{iii}	Codomain	$Ey \rightarrow (ID(cod\ y) \wedge (cod\ y) \cdot y \cong y)$
D_{iii}	Domain	$Ex \rightarrow (ID(dom\ x) \wedge x \cdot (dom\ x) \cong x)$

Categories: Axioms Set II

S_{ii}	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey) \wedge (E(dom\ x) \rightarrow Ex) \wedge (E(cod\ y) \rightarrow Ey)$
E_{ii}	Existence	$E(x \cdot y) \leftarrow (Ex \wedge Ey \wedge (\exists z. z \cdot z \cong z \wedge x \cdot z \cong x \wedge z \cdot y \cong y))$
A_{ii}	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_{ii}	Codomain	$Ey \rightarrow (ID(cod\ y) \wedge (cod\ y) \cdot y \cong y)$
D_{ii}	Domain	$Ex \rightarrow (ID(dom\ x) \wedge x \cdot (dom\ x) \cong x)$

From Monoids to Categories



In Axioms Set III the existence axiom E is simplified by taking advantage of the two new Skolem functions dom and cod.

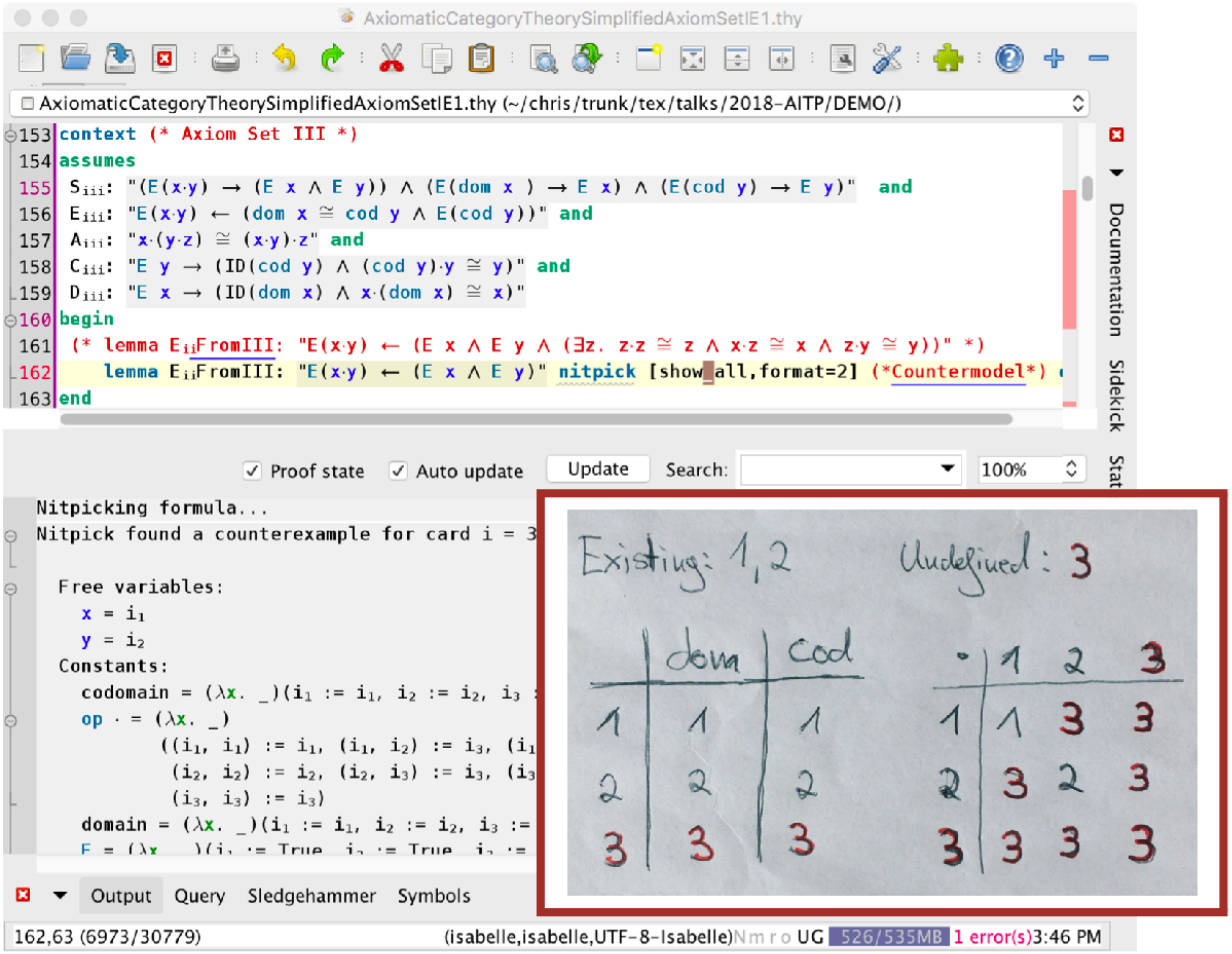
Categories: Axioms Set III

S_{iii}	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey) \wedge (E(dom\ x) \rightarrow Ex) \wedge (E(cod\ y) \rightarrow Ey)$
E_{iii}	Existence	$E(x \cdot y) \leftarrow (dom\ x \cong cod\ y \wedge E(cod\ y))$
A_{iii}	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_{iii}	Codomain	$Ey \rightarrow (ID(cod\ y) \wedge (cod\ y) \cdot y \cong y)$
D_{iii}	Domain	$Ex \rightarrow (ID(dom\ x) \wedge x \cdot (dom\ x) \cong x)$

Experiments with Isabelle/HOL

- Consistency holds (also when $\exists x. \neg(Ex)$): confirmed by **Nitpick**.
- The left-to-right direction of existence axiom E is implied: **Sledgehammer**.
- Axioms Set III implies Axioms Set II: **Sledgehammer**.
- Axioms Set II implies Axioms Set III: **Sledgehammer**.

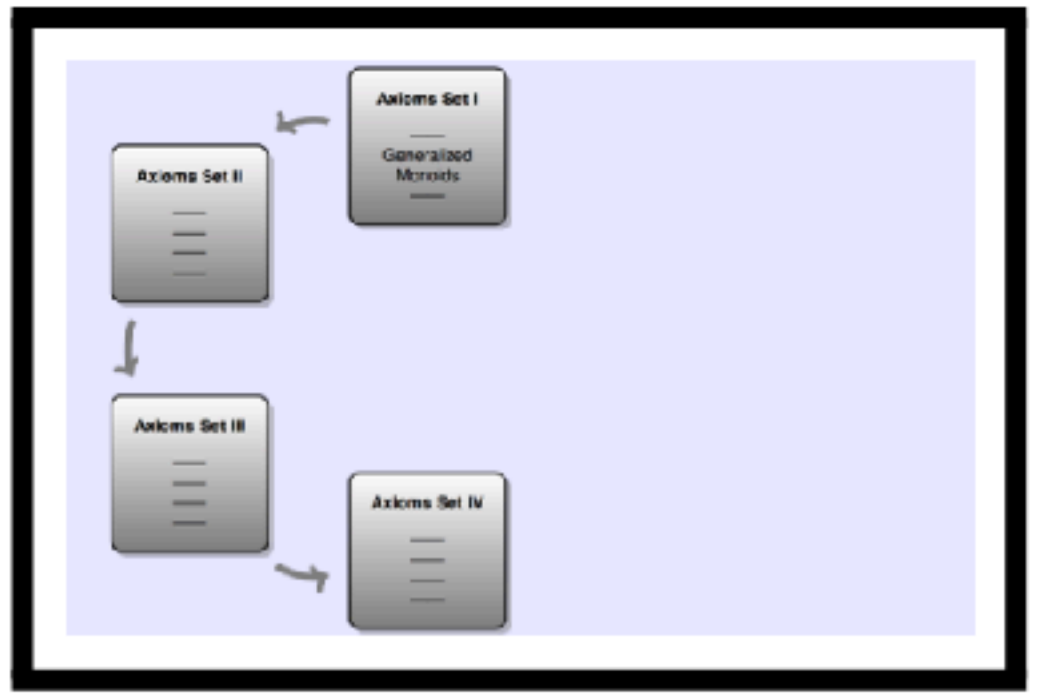
Interesting Model (idempotents, but no left- & right-identities)



	dom	cod
1	1	1
2	2	2
3	3	3

·	1	2	3
1	1	3	3
2	3	2	3
3	3	3	3

From Monoids to Categories



Axioms Set IV simplifies the axioms C and D. However, as it turned out, these simplifications also require the existence axiom E to be strengthened into an equivalence.

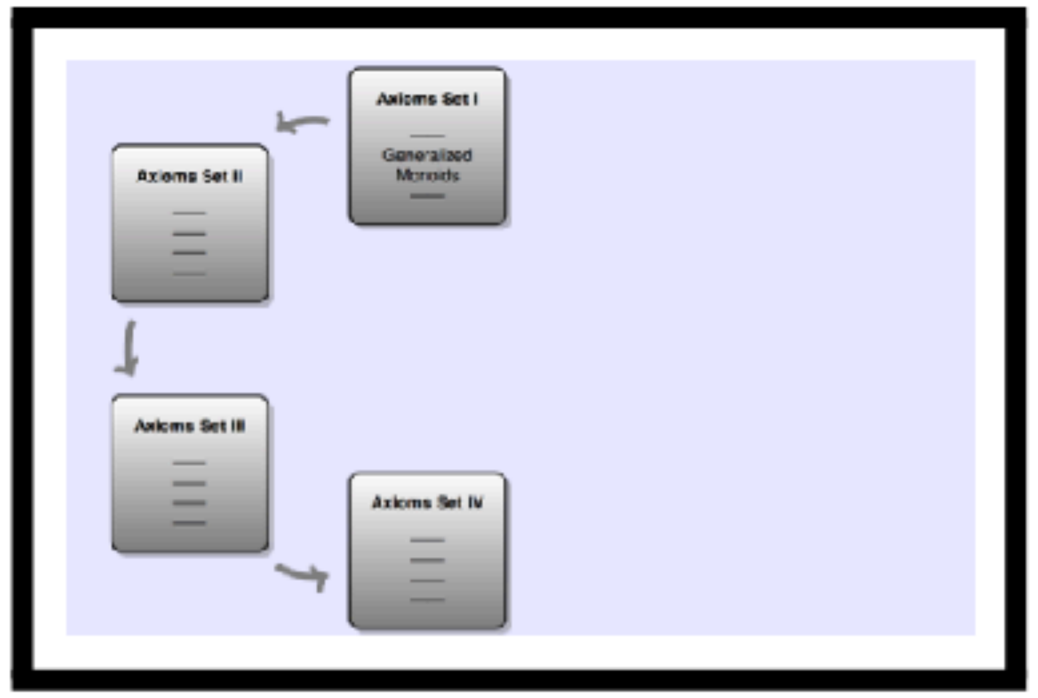
Categories: Axioms Set IV

S_{iv}	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey) \wedge (E(dom\ x) \rightarrow Ex) \wedge (E(cod\ y) \rightarrow Ey)$
E_{iv}	Existence	$E(x \cdot y) \leftrightarrow (dom\ x \cong cod\ y \wedge E(cod\ y))$
A_{iv}	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_{iv}	Codomain	$(cod\ y) \cdot y \cong y$
D_{iv}	Domain	$x \cdot (dom\ x) \cong x$

Categories: Axioms Set III

S_{iii}	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey) \wedge (E(dom\ x) \rightarrow Ex) \wedge (E(cod\ y) \rightarrow Ey)$
E_{iii}	Existence	$E(x \cdot y) \leftarrow (dom\ x \cong cod\ y \wedge E(cod\ y))$
A_{iii}	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_{iii}	Codomain	$Ey \rightarrow (ID(cod\ y) \wedge (cod\ y) \cdot y \cong y)$
D_{iii}	Domain	$Ex \rightarrow (ID(dom\ x) \wedge x \cdot (dom\ x) \cong x)$

From Monoids to Categories



Axioms Set IV simplifies the axioms C and D. However, as it turned out, these simplifications also require the existence axiom E to be strengthened into an equivalence.

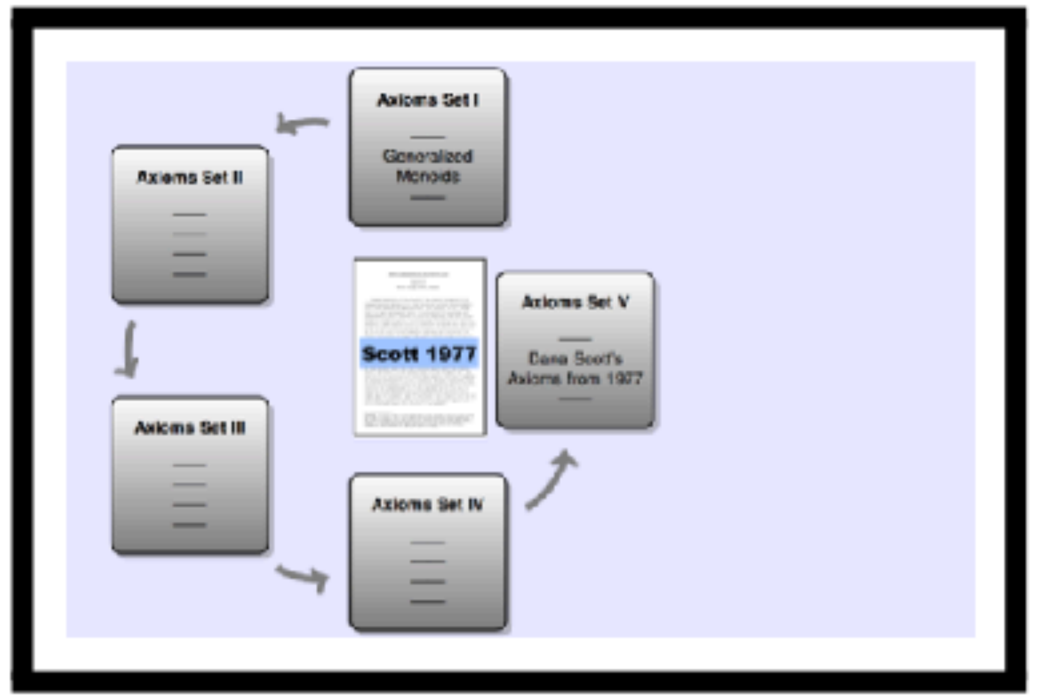
Categories: Axioms Set IV

S_{iv}	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey) \wedge (E(dom\ x) \rightarrow Ex) \wedge (E(cod\ y) \rightarrow Ey)$
E_{iv}	Existence	$E(x \cdot y) \leftrightarrow (dom\ x \cong cod\ y \wedge E(cod\ y))$
A_{iv}	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_{iv}	Codomain	$(cod\ y) \cdot y \cong y$
D_{iv}	Domain	$x \cdot (dom\ x) \cong x$

Experiments with Isabelle/HOL

- Consistency holds (also when $\exists x. \neg(Ex)$): confirmed by **Nitpick**.
- Axioms Set IV implies Axioms Set III: **Sledgehammer**.
- Axioms Set III implies Axioms Set IV: **Sledgehammer**.

From Monoids to Categories



Axioms Set V simplifies axiom E (and S).
Now, strictness of $\cdot$ is implied.

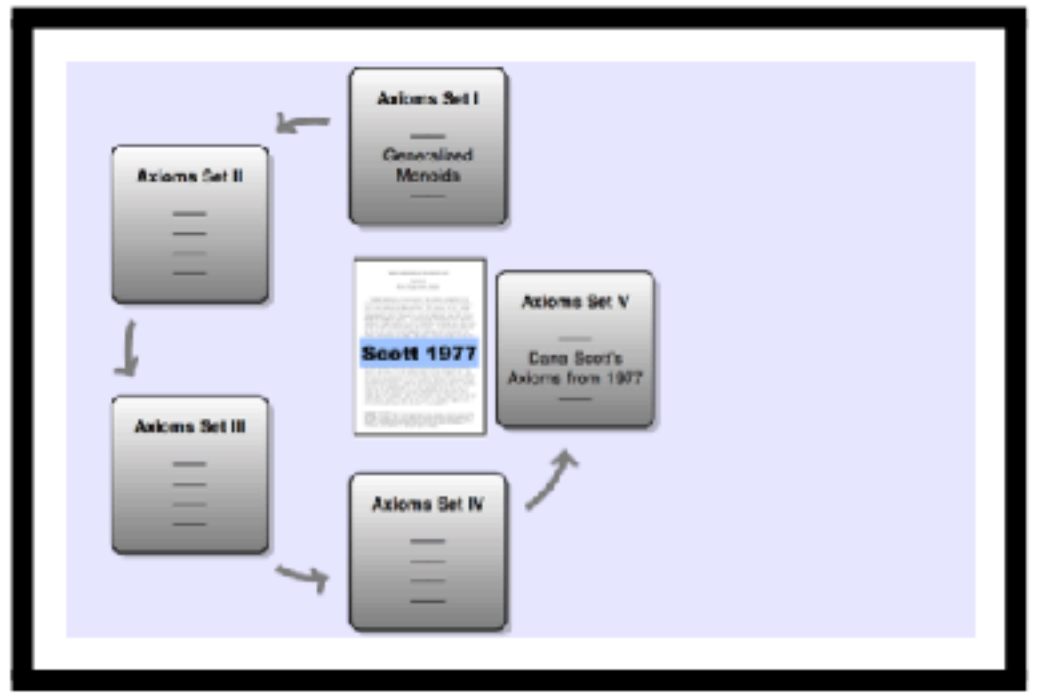
Categories: Axioms Set V (Scott, 1977)

$S1$	Strictness	$E(dom\ x) \rightarrow Ex$
$S2$	Strictness	$E(cod\ y) \rightarrow Ey$
$S3$	Existence	$E(x \cdot y) \leftrightarrow dom\ x \simeq cod\ y$
$S4$	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
$S5$	Codomain	$(cod\ y) \cdot y \cong y$
$S6$	Domain	$x \cdot (dom\ x) \cong x$

Categories: Axioms Set IV

S_{iv}	Strictness	$E(x \cdot y) \rightarrow (Ex \wedge Ey) \wedge (E(dom\ x) \rightarrow Ex) \wedge (E(cod\ y) \rightarrow Ey)$
E_{iv}	Existence	$E(x \cdot y) \leftrightarrow (dom\ x \cong cod\ y \wedge E(cod\ y))$
A_{iv}	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
C_{iv}	Codomain	$(cod\ y) \cdot y \cong y$
D_{iv}	Domain	$x \cdot (dom\ x) \cong x$

From Monoids to Categories



Axioms Set V simplifies axiom E (and S).
Now, strictness of $\cdot$ is implied.

Categories: Axioms Set V (Scott, 1977)

$S1$	Strictness	$E(dom\ x) \rightarrow Ex$
$S2$	Strictness	$E(cod\ y) \rightarrow Ey$
$S3$	Existence	$E(x \cdot y) \leftrightarrow dom\ x \simeq cod\ y$
$S4$	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
$S5$	Codomain	$(cod\ y) \cdot y \cong y$
$S6$	Domain	$x \cdot (dom\ x) \cong x$

Experiments with Isabelle/HOL

- Consistency holds (also when $\exists x.\neg(Ex)$): confirmed by **Nitpick**.
- Axioms Set V implies Axioms Set IV: **Sledgehammer**.
- Axioms Set IV implies Axioms Set V: **Sledgehammer**.

Cats & Alligators

Axioms Set I

Generalized Monoids

Axioms Set II

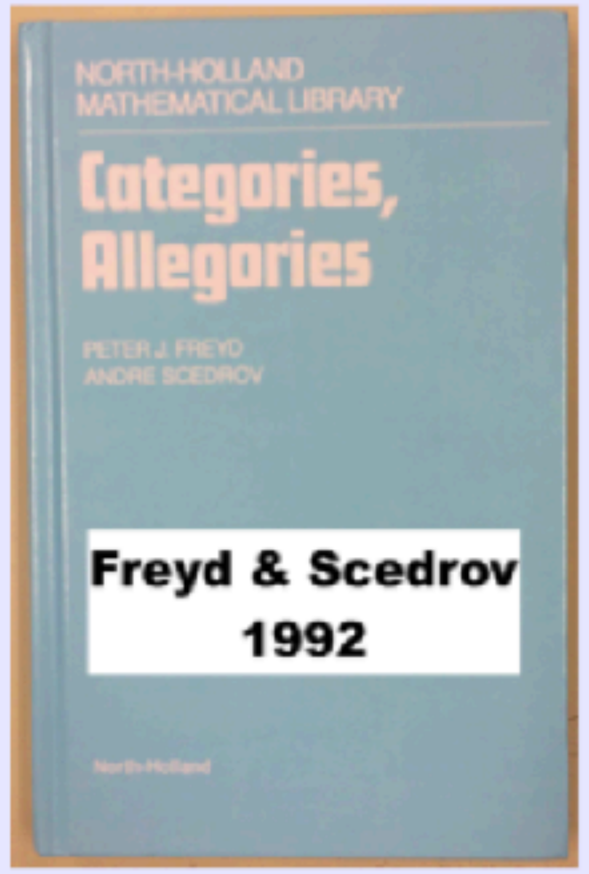
Axioms Set III

Axioms Set IV

Axioms Set V

Dana Scott's Axioms from 1977

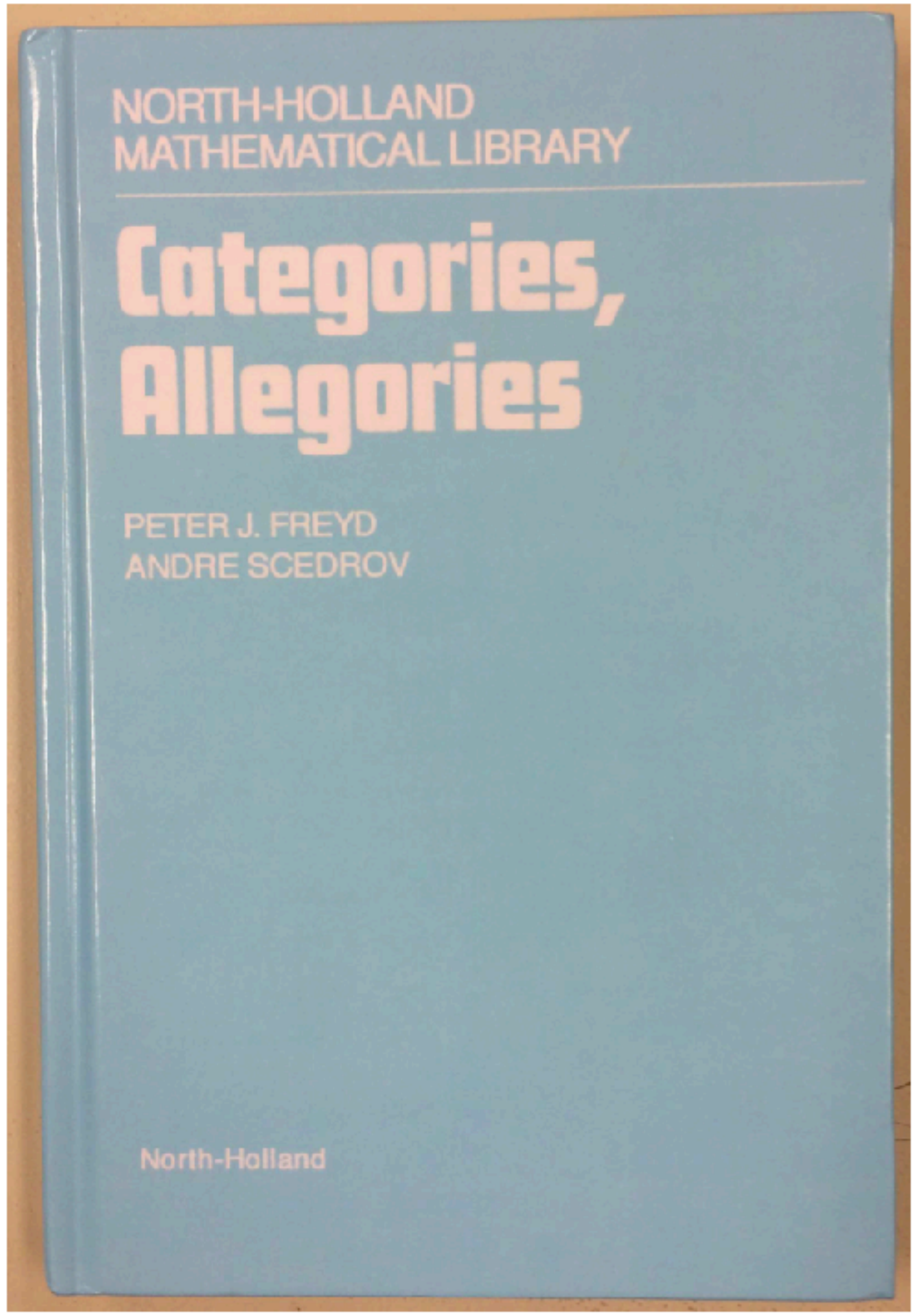
–?–



Axioms Set VI

Freyd & Scedrov's Axioms from 1992

Cats & Alligators



1.1. BASIC DEFINITIONS

The theory of CATEGORIES is given by two unary operations and a binary partial operation. In most contexts lower-case variables are used for the 'individuals' which are called *morphisms* or *maps*. The values of the operations are denoted and pronounced as:

$\Box x$ *the source of* x ,

$x\Box$ *the target of* x ,

xy *the composition of* x *and* y .

The axioms:

A1 xy *is defined iff* $x\Box = \Box y$,

A2a $(\Box x)\Box = \Box x$ *and* $\Box(x\Box) = x\Box$, A2b

A3a $(\Box x)x = x$ *and* $x(x\Box) = x$, A3b

A4a $\Box(xy) = \Box(x(\Box y))$ *and* $(xy)\Box = ((x\Box)y)\Box$, A4b

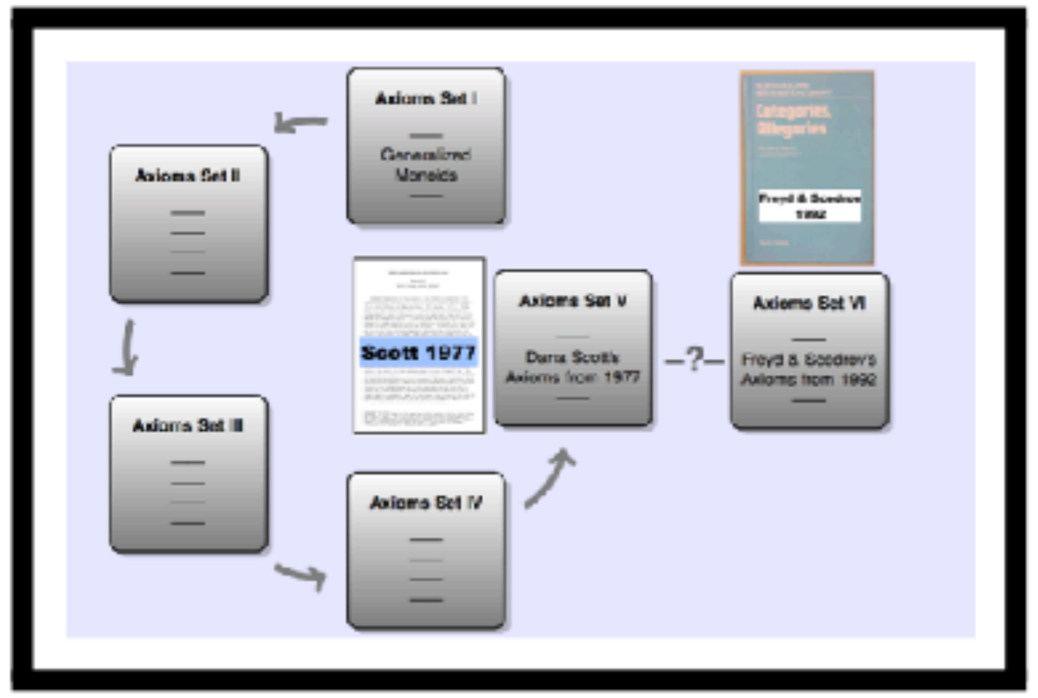
A5 $x(yz) = (xy)z$.

1.11. The ordinary equality sign = will be used only in the symmetric sense, to wit: if either side is defined then so is the other and they are equal. A theory, such as this, built on an ordered list of partial operations, the domain of definition of each given by equations in the previous, and with all other axioms equational, is called an ESSENTIALLY ALGEBRAIC THEORY.

1.12. We shall use a venturi-tube $\succeq$ for *directed equality* which means: if the left side is defined then so is the right and they are equal. The axiom that $\Box(xy) = \Box(x(\Box y))$ is equivalent, in the presence of the earlier axioms, with $\Box(xy) \succeq \Box x$ as can be seen below.

1.13. $\Box(\Box x) = \Box x$ because $\Box(\Box x) = \Box((\Box x)\Box) = (\Box x)\Box = \Box x$. Similarly $(x\Box)\Box = x\Box$.

Cats & Alligators

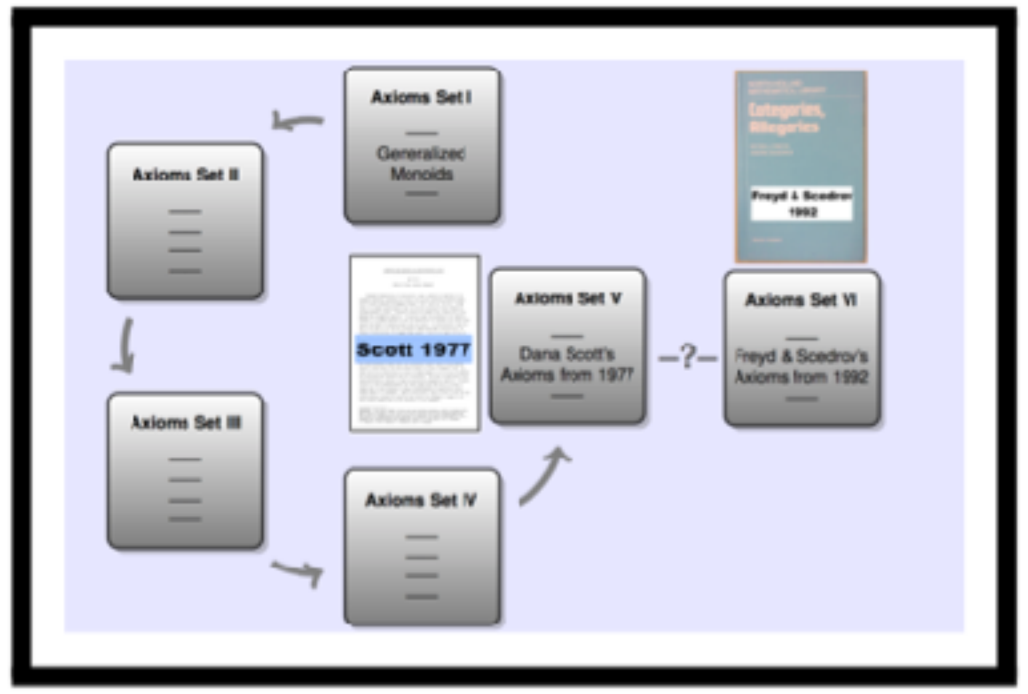


Categories: Original axiom set by Freyd and Scedrov (modulo notation)

A1	$E(x \cdot y) \leftrightarrow dom\ x \ \cong cod\ y$
A2a	$cod(dom\ x) \cong dom\ x$
A2b	$dom(cod\ y) \cong cod\ y$
A3a	$x \cdot (dom\ x) \cong x$
A3b	$(cod\ y) \cdot y \cong y$
A4a	$dom(x \cdot y) \cong dom((dom\ x) \cdot y)$
A4b	$cod(x \cdot y) \cong cod(x \cdot (cod\ y))$
A5	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$

Experiments with Isabelle/HOL

- Consistency? — Nitpick finds a model.
- Consistency when assuming $\exists x. \neg Ex$ — Nitpick does not find a model.
- lemma $(\exists x. \neg Ex) \rightarrow False$: **Sledgehammer**. (Problematic axioms: $A1, A2a, A3a$)



Categories: Axioms Set VI (Freyd and Scedrov, when corrected)

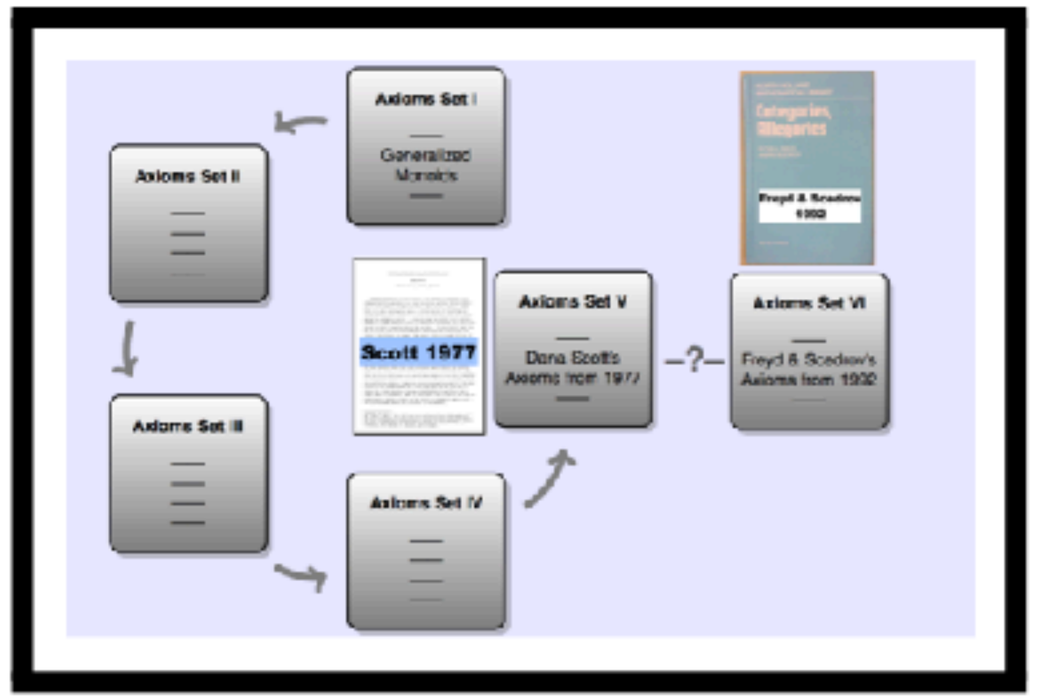
A1	$E(x \cdot y) \leftrightarrow dom\ x \simeq cod\ y$
A2a	$cod(dom\ x) \cong dom\ x$
A2b	$dom(cod\ y) \cong cod\ y$
A3a	$x \cdot (dom\ x) \cong x$
A3b	$(cod\ y) \cdot y \cong y$
A4a	$dom(x \cdot y) \cong dom((dom\ x) \cdot y)$
A4b	$cod(x \cdot y) \cong cod(x \cdot (cod\ y))$
A5	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$

Experiments with Isabelle/HOL

- Consistency holds (also when $\exists x.\neg(Ex)$): confirmed by **Nitpick**.
- Axioms Set VI implies Axioms Set V: **Sledgehammer**.
- Axioms Set V implies Axioms Set VI: **Sledgehammer**.
- Redundancies:

— The $A4$-axioms are implied by the others: **Sledgehammer**.
— The $A2$-axioms are implied by the others: **Sledgehammer**.

Cats & Alligators



Maybe Freyd and Scedrov do not assume a free logic. In algebraic theories free variables often range over existing objects only. However, we can formalise this as well:

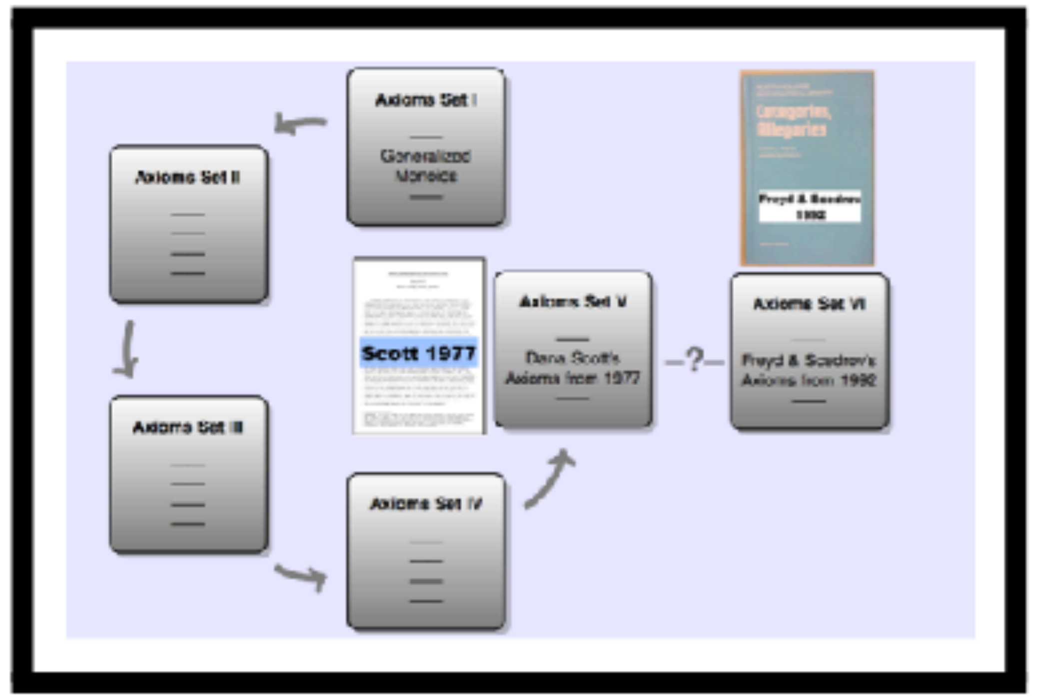
Categories: "Algebraic reading" of axiom set by Freyd and Scedrov.

A1	$\forall xy.\ E(x \cdot y) \leftrightarrow dom\ x \cong cod\ y$
A2a	$\forall x.\ cod(dom\ x) \cong dom\ x$
A2b	$\forall y.\ dom(cod\ y) \cong cod\ y$
A3a	$\forall x.\ x \cdot (dom\ x) \cong x$
A3b	$\forall y.\ (cod\ y) \cdot y \cong y$
A4a	$\forall xy.\ dom(x \cdot y) \cong dom((dom\ x) \cdot y)$
A4b	$\forall xy.\ cod(x \cdot y) \cong cod(x \cdot (cod\ y))$
A5	$\forall xyz.\ x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$

Experiments with Isabelle/HOL

- Consistency holds (also when $\exists x. \neg(Ex)$): confirmed by **Nitpick**.
- However, none of V-axioms are implied: **Nitpick**.
- For equivalence to V-axioms: add strictness of dom, cod, $\cdot$, **Sledgehammer**.

Cats & Alligators



Maybe Freyd and Scedrov do not assume a free logic. In algebraic theories free variables often range over existing objects only. However, we can formalise this as well:

Categories: "Algebraic reading" of axiom set by Freyd and Scedrov.

A1	$\forall xy.\ E(x \cdot y) \leftrightarrow dom\ x \cong cod\ y$
A2a	$\forall x.\ cod(dom\ x) \cong dom\ x$
A2b	$\forall y.\ dom(cod\ y) \cong cod\ y$
A3a	$\forall x.\ x \cdot (dom\ x) \cong x$
A3b	$\forall y.\ (cod\ y) \cdot y \cong y$
A4a	$\forall xy.\ dom(x \cdot y) \cong dom((dom\ x) \cdot y)$
A4b	$\forall xy.\ cod(x \cdot y) \cong cod(x \cdot (cod\ y))$
A5	$\forall xyz.\ x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$

Experiments with Isabelle/HOL

But: Strictness is not mentioned in Freyd and Scedrov!
And it could not even be expressed axiomatically, when variables range over of existing objects only. This leaves us puzzled about their axiom system.

Hence, we better prefer the Axioms Set V by Scott (from 1977).

GROUPS, CATEGORIES AND DUALITY

By Saunders MacLane*

Department of Mathematics, University of Chicago

Communicated by Marshall Stone, May 1, 1948

It has long been recognized that the theorems of group theory display a certain duality. The concept of a lattice gives a partial expression for this duality, in that some of the theorems about groups which can be formulated in terms of the lattice of subgroups of a group display the customary lattice duality between meet (intersection) and join (union). The duality is not always present, in the sense that the lattice dual of a true theorem on groups need not be true; for example, a Jordan Holder theorem holds for certain ascending well-ordered infinite composition series, but not for the corresponding descending series.[1] Moreover, there are other striking group theoretic situations where a duality is present, but is not readily expressible in lattice-theoretic terms.

As an example, consider the direct product $D = G \times H$ of two groups

GROUPS, CATEGORIES AND DUALITY

BY SAUNDERS MACLANE*

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF CHICAGO

Communicated by Marshall Stone, May 1, 1948

It has long been recognized that the theorems of group theory display a certain duality. The concept of a lattice gives a partial expression for this duality, in that some of the theorems about groups which can be formulated in terms of the lattice of subgroups of a group display the customary lattice duality between meet (intersection) and join (union). The duality is not always present, in the sense that the lattice dual of a true the
theorem
series, bu
are othe
but is no
As an

introduced the notion of a category.[6] A *category* is a class of "mappings" (say, homomorphisms) in which the product $\alpha\beta$ of certain pairs of mappings α and β is defined. A mapping e is called an *identity* if $\rho\alpha = \alpha$ and $\beta\rho = \beta$ whenever the products in question are defined. These products must satisfy the axioms:

(C-1). *If the products $\gamma\beta$ and $(\gamma\beta)\alpha$ are defined, so is $\beta\alpha$;*

(C-1′). *If the products $\beta\alpha$ and $\gamma(\beta\alpha)$ are defined, so is $\gamma\beta$;*

(C-2). *If the products $\gamma\beta$ and $\beta\alpha$ are defined, so are the products $(\gamma\beta)\alpha$ and $\gamma(\beta\alpha)$, and these products are equal.*

(C-3). *For each γ there is an identity e_D such that γe_D is defined;*

(C-4). *For each γ there is an identity e_R such that $e_R\gamma$ is defined.*

It follows that the identities e_D and e_R are unique; they may be called, respectively, the *domain* and the *range* of the given mapping γ. A mapping θ with a two-sided inverse is an *equivalence*.

These axioms are clearly self dual, and a dual theory of free and direct products may be constructed in any category in which such products exist.

As before, we adopt an algebraic reading and add an explicit strictness condition.

Categories: Axioms Set by Mac Lane

C0 $E(\gamma \cdot \beta) \rightarrow (E\gamma \wedge E\beta)$ **(added by us)**

C1 $\forall \gamma, \beta, \alpha.\ (E(\gamma \cdot \beta) \wedge E((\gamma \cdot \beta) \cdot \alpha)) \rightarrow E(\beta \cdot \alpha)$

C1' $\forall \gamma, \beta, \alpha.\ (E(\beta \cdot \alpha) \wedge E(\gamma \cdot (\beta \cdot \alpha)) \rightarrow E(\gamma \cdot \beta)$

C2 $\forall \gamma, \beta, \alpha.\ (E(\gamma \cdot \beta) \wedge E(\beta \cdot \alpha)) \rightarrow$
$(E((\gamma \cdot \beta) \cdot \alpha) \wedge E(\gamma \cdot (\beta \cdot \alpha)) \wedge ((\gamma \cdot \beta) \cdot \alpha) = (\gamma \cdot (\beta \cdot \alpha)))$

C3 $\forall \gamma.\ \exists eD.\ IDMcL(eD) \wedge E(\gamma \cdot eD)$

C4 $\forall \gamma.\ \exists eR.\ IDMcL(eR) \wedge E(eR \cdot \gamma)$

where $IDMcL(\rho) \equiv (\forall \alpha.\ E(\rho \cdot \alpha) \rightarrow \rho \cdot \alpha = \alpha) \wedge (\forall \beta.\ E(\beta \cdot \rho) \rightarrow \beta \cdot \rho = \beta)$

Consistency holds (also when $\exists x. \neg(Ex)$): confirmed by **Nitpick**.

How about the Skolemized variant?

Categories: Axioms Set by Mac Lane

C0	$(E(\gamma \cdot \beta) \to (E\gamma \wedge E\beta)) \wedge (E(dom\ \gamma) \to (E\gamma)) \wedge (E(cod\ \gamma) \to (E\gamma))$	**(added)**
C1	$\forall \gamma, \beta, \alpha.\ (E(\gamma \cdot \beta) \wedge E((\gamma \cdot \beta) \cdot \alpha)) \to E(\beta \cdot \alpha)$	
C1'	$\forall \gamma, \beta, \alpha.\ (E(\beta \cdot \alpha) \wedge E(\gamma \cdot (\beta \cdot \alpha)) \to E(\gamma \cdot \beta)$	
C2	$\forall \gamma, \beta, \alpha.\ (E(\gamma \cdot \beta) \wedge E(\beta \cdot \alpha)) \to$ $(E((\gamma \cdot \beta) \cdot \alpha) \wedge E(\gamma \cdot (\beta \cdot \alpha)) \wedge ((\gamma \cdot \beta) \cdot \alpha) = (\gamma \cdot (\beta \cdot \alpha)))$	
C3	$\forall \gamma.\ IDMcL(dom\ \gamma) \wedge E(\gamma \cdot (dom\ \gamma))$	
C4	$\forall \gamma.\ IDMcL(cod\ \gamma) \wedge E((cod\ \gamma) \cdot \gamma)$	

Consistency holds (also when $\exists x. \neg(Ex)$): confirmed by **NITPICK**.

This axioms set is equivalent to (as shown by Sledgehammer)

Categories: Axioms Set V (Scott, 1977)

$S1$	Strictness	$E(dom\ x) \to Ex$
$S2$	Strictness	$E(cod\ y) \to Ey$
$S3$	Existence	$E(x \cdot y) \leftrightarrow dom\ x \simeq cod\ y$
$S4$	Associativity	$x \cdot (y \cdot z) \cong (x \cdot y) \cdot z$
$S5$	Codomain	$(cod\ y) \cdot y \cong y$
$S6$	Domain	$x \cdot (dom\ x) \cong x$

How about the Skolemized variant?

Categories: Axioms Set by Mac Lane

C0	$(E(\gamma \cdot \beta) \rightarrow (E\gamma \wedge E\beta)) \wedge (E(dom\ \gamma) \rightarrow (E\gamma)) \wedge (E(cod\ \gamma) \rightarrow (E\gamma))$	**(added)**
C1	$\forall \gamma, \beta, \alpha.\ (E(\gamma \cdot \beta) \wedge E((\gamma \cdot \beta) \cdot \alpha)) \rightarrow E(\beta \cdot \alpha)$	
C1'	$\forall \gamma, \beta, \alpha.\ (E(\beta \cdot \alpha) \wedge E(\gamma \cdot (\beta \cdot \alpha)) \rightarrow E(\gamma \cdot \beta)$	
C2	$\forall \gamma, \beta, \alpha.\ (E(\gamma \cdot \beta) \wedge E(\beta \cdot \alpha)) \rightarrow$ $(E((\gamma \cdot \beta) \cdot \alpha) \wedge E(\gamma \cdot (\beta \cdot \alpha)) \wedge ((\gamma \cdot \beta) \cdot \alpha) = (\gamma \cdot (\beta \cdot \alpha)))$	
C3	$\forall \gamma.\ IDMcL(dom\ \gamma) \wedge E(\gamma \cdot (dom\ \gamma))$	
C4	$\forall \gamma.\ IDMcL(cod\ \gamma) \wedge E((cod\ \gamma) \cdot \gamma)$	

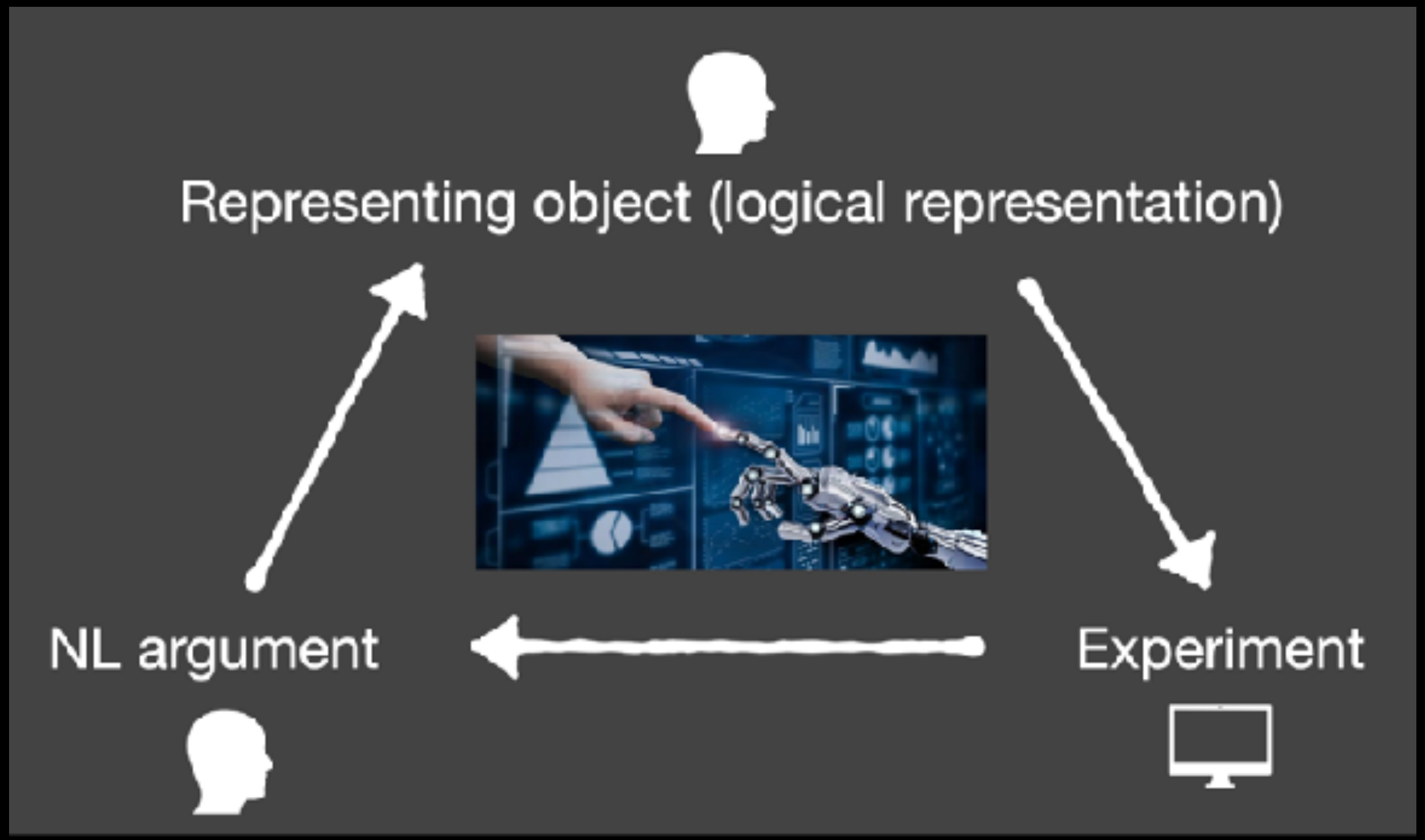
Consistency holds (also when $\exists x. \neg(Ex)$): confirmed by **Nitpick**.

See also our "Archive of Formal Proofs" entry at:
`https://www.isa-afp.org/entries/AxiomaticCategoryTheory.html`

Results of this study (using free logic in HOL)

Axiom Systems for Category Theory

- Connection depicted to generalised monoids
- Minimal axiom systems, dependencies
- Consistency, strictness assumptions
- Mutual relationships explored



Methological Results

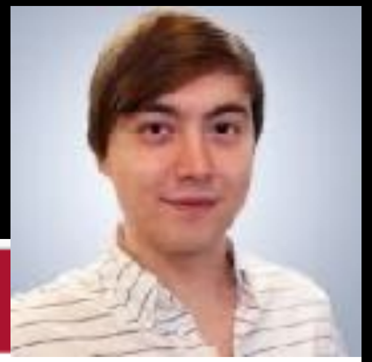
- Evidence for LogiKEy methodology when applied to free logic
- High degree of automation: theorem proving & (counter-)model finding
- Required familiarity with Isabelle/HOL still (too) high for non-experts

Obvious Question

- How about digging deeper?

Further Experiments

lucca@tiemens.de



International Conference on Relational and Algebraic Methods in Computer Science
↳ RAMiCS 2020: **Relational and Algebraic Methods in Computer Science** pp 302–317

Computer-Supported Exploration of a Categorical Axiomatization of Modeloids

Lucca Tiemens, Dana S. Scott, Christoph Benzmüller & Miroslav Benda

A **modeloid** abstracts from a structure to the set of its partial automorphisms.

Using our axiomatisation of category theory we develop a generalization of a modeloid first to an inverse semigroup and then to an **inverse category**.

Formal framework to study relationship between structures of same vocabulary.

Abstract representation of Ehrenfeucht-Fraisse games between two structures.

gonus.aleksey@gmail.com

Freie Universität Berlin

Categorical semantics of Intuitionistic Multiplicative Linear Logic and its formalization in Isabelle/HOL

Master's thesis

17 May 2021

Focuses on fragment of linear logic: **intuitionistic multiplicative LL (IMLL)**; further generalisation possible.

Using our axiomatisation of category theory an interpretation of IMLL formulas and rules in **symmetric monoidal closed categories** is presented.

Sound Modeling & Automation: IMLL modelled in Axiomatic Category Theory modelled in Free Logic modell. in HOL.

jonas.bayer@fu-berlin.de

FREIE UNIVERSITÄT BERLIN

BACHELOR'S THESIS

Exploring categories, formally

2021

JONAS BAYER

Studies **practicability/elegance** of axiomatic category theory approach.

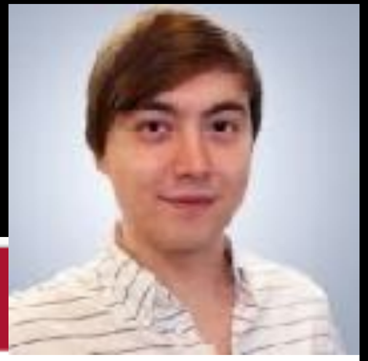
Studies i**nfinite structures:** category α-Set of functions between sets (with α-type elements); good automation.

Categ. with products & coproducts; some limitations discussed.

Category of categories: proves that categories themselves form a category with functors as arrows.

Further Experiments

lucca@tiemens.de



International Conference on Relational and Algebraic Methods in Computer Science
↳ RAMiCS 2020: Relational and Algebraic Methods in Computer Science pp 302–317

Computer-Supported Exploration of a Cate
Axiomatization of Modeloids

Lucca Tiemens, Dana S. Scott, Christoph Benzmüller & Miroslav Benda

gonus.aleksey@gmail.com

Freie Universität Berlin

Categorical semantics of Intuitionistic Multiplicative Linear Logic and its formalization in Isabelle/HOL

jonas.bayer@fu-berlin.de

FREIE UNIVERSITÄT BERLIN

BACHELOR'S THESIS

Exploring categories, formally

A **modeloid** abstracts from a
to the set of its partial automo

Using our axiomatisation of theory we develop a generali
modeloid first to an inverse se
and then to an **inverse categ**

Formal framework to study re
between structures of same v

Abstract representation of Ehrenfeucht-Fraisse games
two structures.

Further formalized concepts

Constructions	Instantiations	Categories + Structure
(Co)products	(typed) category Set	+ Binary (co)product
Equalizers	Category of Posets	Cartesian categories
Final & initial objects	Binary (co)product Category of Lattices	Cartesian closed categories
Exponentials		
Limits (generically)	Category of Categories	Elementary Toposes
Pullbacks		

e of
oach.
egory
s (with
tion.
ducts;
s that
s.

Free Higher-Order Logic

Free Higher-order Logic

Freie Universität Berlin

Free Higher-Order Logic

Notion, Definition and Embedding in HOL

Master's Thesis

March 10, 2020

SpringerLink Search

German Conference on Artificial Intelligence (Künstliche Intelligenz)

KI 2020: KI 2020: Advances in Artificial Intelligence pp 116-131 | Cite as

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

Authors Authors and affiliations

Irina Makarenko, Christoph Benzmüller

Conference paper

First Online: 09 September 2020

476 Downloads

https://github.com/stilleben/Free-Higher-Order-Logic

Freie Universität Berlin

Positive Free Higher-Order Logic

and its Automation via a Semantical Embedding

Irina Makarenko and Christoph Benzmüller

KI'2020 · Knowledge Representation and Reasoning
24.09.2020

Contributions of Irina

- Basic Notion of **free HOL** adapting prior work by (Farmer 1990, 1993, 2004)

- Definition & encoding of variants of **positive** free HOL, including different constraints regarding existence of Booleans and strictness
  Positive: atomic formulas with non-denoting terms may evaluate to true (Lambert 1963, 1967; Scott 1967)

- Definition & encoding of variants of **negative** free HOL, including …
  Negative: atomic formulas with non-denoting terms evaluate to false (Schock 1964, 1968; Scales 1969; Burge 1974)

- Definition of variants of **neutral** free HOL
  Neutral: atomic formulas with non-denoting terms evaluate to indetermined truth value (Lehmann 1994, 2001, 2002)

- Definition of variants of free HOL with **supervaluation semantics**
  Supervaluation: partial valuation function is extended to a total one by by considering all the values that the subformulas of a formula could have if their empty terms had referents. (Frassen 1966, Skryms 1968, Meyer and Lambert 1968, Bencivenga 1981, 1986)

Contributions of Irina (cont'd)

- Experiments in Isabelle/HOL using Prior's paradox as running example:
  - Positive free HOL: True if (E True) and (E False) are assumed, False otherwise
  - Negative free HOL: False
- Alternatives regarding base choices can be studied
- Experiments with supervaluation semantics (exploiting Barba Escriba's (2001) sound and complete translation into negative free modal logic S4.1)
- Irina's thesis and the Isabelle/HOL sources are available online:
  - https://www.mi.fu-berlin.de/inf/groups/ag-ki/Theses/Completed-theses/Master_Diploma-theses/2020/Makarenko/MA-Makarenko.pdf
  - https://github.com/stilleben/Free-Higher-Order-Logic

Another very complex study where Free Higher-Order Logic plays a role

Further Foundational Studies: Metaphysical Theory

(PhD of Daniel Kirchner, supervised by Ed Zalta and myself)

Principia Logico-Metaphysica (Draft/Excerpt)

Edward N. Zalta
Philosophy Department
Stanford University

With critical theoretical contributions by

Daniel Kirchner
AI Systems Engineering
Universität Bamberg
and
Uri Nodelman
Philosophy Department
Stanford University

May 22, 2024

https://mally.stanford.edu/principia.pdf

According to clause (.6.a), any formula φ can serve as the matrix of a relation term of the form $[\lambda \nu_1 \dots \nu_n\, \varphi]$, where $n \geq 0$. If we allow ourselves to speak informally about the denotation of a term, then it is important to alert the reader to the following facts about the system we shall be developing:

- Not every λ-expression is guaranteed to have a denotation (indeed, some λ-expressions will provably fail to have denotations), and so these expressions, like definite descriptions, will be governed by a negative free logic.
- Every 0-ary λ-expression $[\lambda\, \varphi]$ is guaranteed to have a denotation, by axiom (39.2), and every formula φ is guaranteed to have a denotation, by theorem (104.2).
- It will be provable that $[\lambda \varphi]$ and φ always denote the same 0-ary relation, by theorem (111.1).

In general, λ-expressions are *not* to be interpreted as terms that potentially denote functions, but rather as terms that potentially denote relations. Thus, when we introduce axioms and rules of inference governing λ-expressions in the next two chapters, the resulting λ-calculus is to be understood as a calculus of relations.

Further Foundational Studies: Metaphysical Theory

(PhD of Daniel Kirchner, supervised by Ed Zalta and myself)

Principia Logico-Metaphysica
(Draft/Excerpt)

Edward N. Zalta
Philosophy Department
Stanford University

With critical theoretical contributions by

Daniel Kirchner
AI Systems Engineering
Universität Bamberg
and
Uri Nodelman
Philosophy Department
Stanford University

May 22, 2024

https://mally.stanford.edu/principia.pdf

Computer-Verified Foundations of Metaphysics
and an Ontology of Natural Numbers in Isabelle/HOL

Dissertation
zur Erlangung des Grades eines
Doktors der Naturwissenschaften

am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

vorgelegt von

Daniel Kirchner

Berlin, December 2021

Entire PhD thesis was written directly in Isabelle/HOL

Further Foundational Studies: Metaphysical Theory

(PhD of Daniel Kirchner, supervised by Ed Zalta and myself)

Principia Logico-Metaphysica
(Draft/Excerpt)

Edward N. Zalta
Philosophy Department
Stanford University

With critical theoretical contributions by

Daniel Kirchner
AI Systems Engineering
Universität Bamberg
and
Uri Nodelman
Philosophy Department
Stanford University

May 22, 2024

https://mally.stanford.edu/principia.pdf

Foundational metaphysical theory (based on a hyperintensional relational HO modal logic)

Formalised & studied in Isabelle/HOL

- approx. 24000 loc
- using LogiKEy methodology
- paradox rediscovered & fixed
- derivation of natural numbers

Latest versions of this theory shifted towards free logic; strongly influenced (& verified) by computer-experiments

Further Foundational Studies: Metaphysical Theory

(PhD of Daniel Kirchner, supervised by Ed Zalta and myself)

Principia Logico-Metaphysica
(Draft/Excerpt)

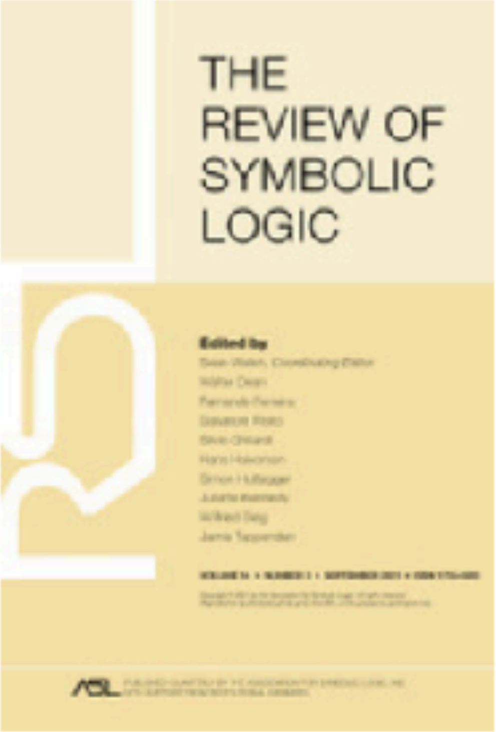
Edward N. Zalta
Philosophy Department
Stanford University

With critical theoretical contributions by

Daniel Kirchner
AI Systems Engineering
Universität Bamberg
and
Uri Nodelman
Philosophy Department
Stanford University

May 22, 2024

https://mally.stanford.edu/principia.pdf

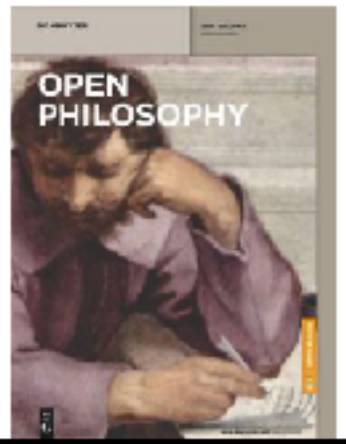


MECHANIZING *PRINCIPIA LOGICO-METAPHYSICA* IN FUNCTIONAL TYPE-THEORY

Published online by Cambridge University Press: **12 July 2019**

DANIEL KIRCHNER, CHRISTOPH BENZMÜLLER and EDWARD N. ZALTA

Show author details



ARTICLE Open Access

Computer Science and Metaphysics: A Cross-Fertilization

Open Philosophy

Daniel Kirchner, Christoph Benzmüller, Edward N. Zalta August 23, 2019

Isabelle/HOL Code (~24000 loc):
https://github.com/ekpyron/AOT

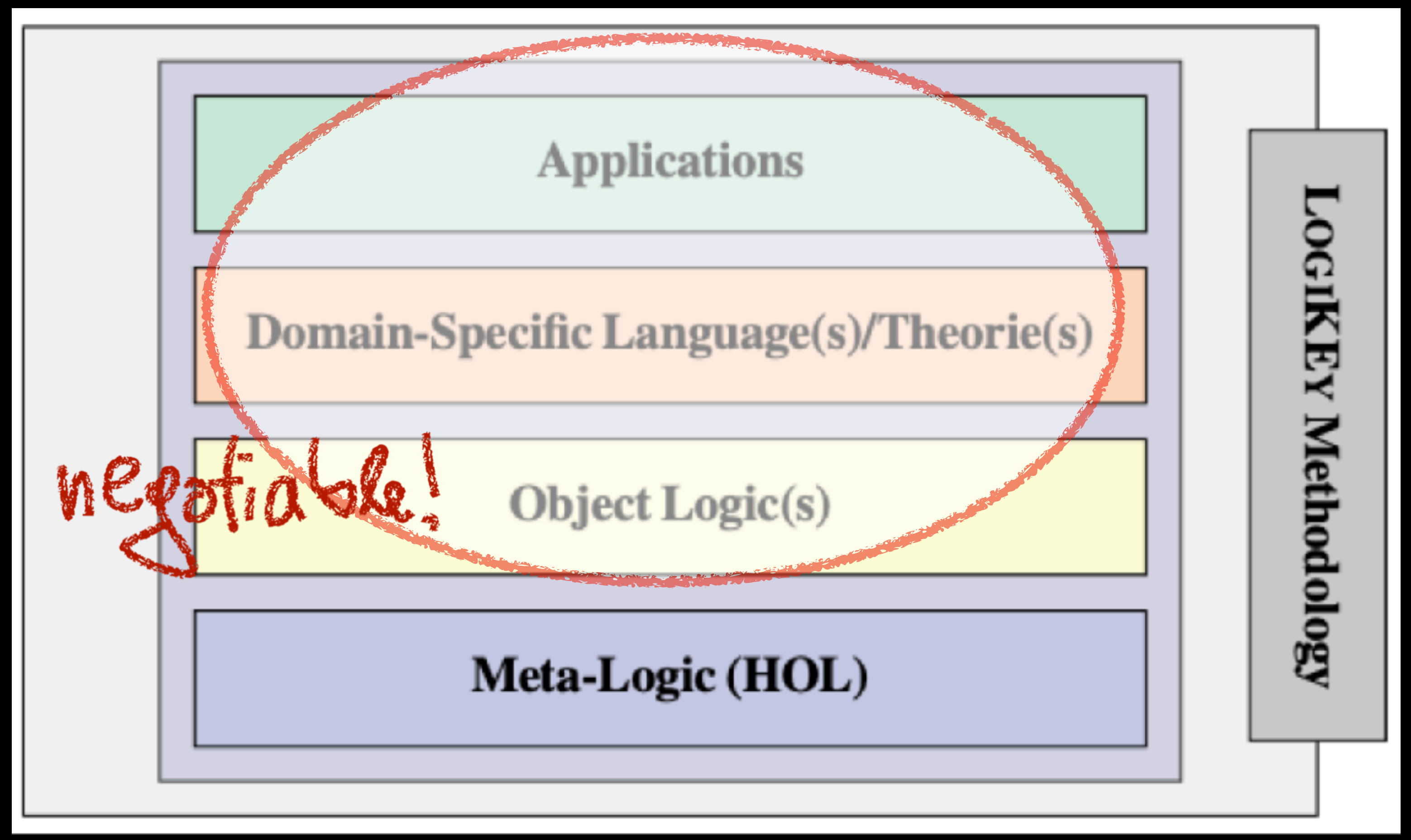
daniel@ekpyron.org

Conclusion I: Successful Application(s) of LogiKEy

Human-Computer Interaction

Revision (often small changes):

Representing object (logical representation)



Modified experiments:

Revision:

Argument/Theory

Experiment

New insights (e.g. falsification)

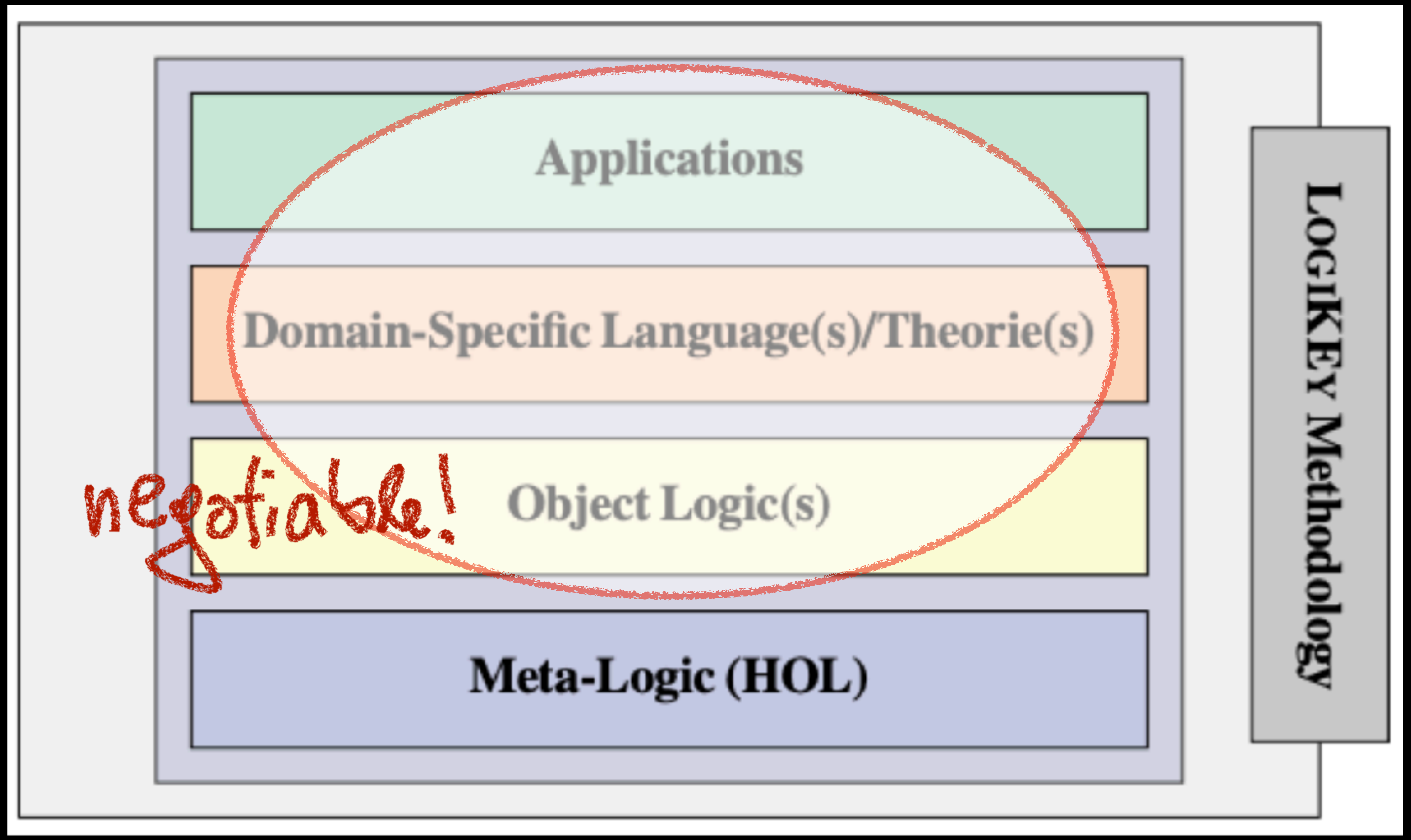
Conclusion I: Logico-Pluralistic LogiKEy Approach

LogiKEy successfully applied for

- a wide range of object logics
- various object logic combinations
- different application domains (with contribution of new insights)

LogiKEy in Isabelle/HOL

- good proof automation with Sledgehammer
- even more valuable is (counter-)model finding with Nitpick
- very good syntax representations



LogiKEy offers a uniform methodology and infrastructure where even object logics and their conbinations become negotiable and objects of study.

Conclusion II: Free Logic in LogiKEy

- Free logics can be explored as fragments of HOL
- Elegant shallow embeddings using the LogiKEy methodology enable …
- … intuitive user interaction within a proof assistant such as Isabelle/HOL
- … proof automation and (counter-)model finding in Isabelle/HOL
- Applications (also larger ones) are well supported and …
- … errors/issues can be detected

Let's end the wallflower existence of free logics and explore and foster its adoption in practical applications!

Reading

(preprints: http://christoph-benzmueller.de/publications.html)

Free Logic & Axiomatic Category Theory in Free Logic

- Scott (1967), **Existence and description in formal logic.** Reprinted in: Lambert (1991), Philosophical Application of Free Logic, OUP. (ISBN 0195061314)
- Scott (1977), **Identity and existence in intuitionistic logic.** Applications of Sheaves, Springer (https://doi.org/10.1007/BFb0061839)
- Benzmüller & Scott (2016), **Automating Free Logic in Isabelle/HOL.** ICMS 2016 (https://doi.org/10.1007/978-3-319-42432-3_6)
- __ (2018), **Axiom systems for category theory in free logic**. Archive of Formal Proofs (https://www.isa-afp.org/entries/AxiomaticCategoryTheory.html)
- __ (2020), **Automating Free Logic in HOL, with an Experimental Application in Category Theory.** JAR (http://doi.org/10.1007/s10817-018-09507-7)
- Tiemens, Scott, Benzmüller & Benda (2020), **Computer-supported Exploration of a Categorical Axiomatization of Modeloids.** RAMiCS 2020 (https://doi.org/10.1007/978-3-030-43520-2_19)
- Makarenko & Benzmüller (2020), **Positive Free Higher-Order Logic and its Automation via a Semantical Embedding**. KI 2020 (http://doi.org/10.1007/978-3-030-58285-2_9)
- Bayer (2021), **Exploring categories, formally**. BSc Thesis, Dep. of Maths and CS, FU Berlin.
- Gonus (2021), **Categorical semantics of Intuitionistic Multiplicative Linear Logic and its formalization in Isabelle/HOL**. MSc Thesis, Dep. of Maths and CS, FU Berlin.

Reading

(preprints: http://christoph-benzmueller.de/publications.html)

HOL (see also the papers of Church, Henkin, Andrews and Muskens in JSL as mentioned before)

- Benzmüller, Brown, Kohlhase (2004), Higher-Order Semantics and Extensionality, J. Symb. Log., http://doi.org/10.2178/jsl/1102022211
- Benzmüller & Miller (2014), Automation of Higher-Order Logic, Handbook of the History of Logic, http://doi.org/10.1016/B978-0-444-51624-4.50005-8
- Benzmüller & Andrews (2019), Church's Type Theory, SEP, https://plato.stanford.edu/entries/type-theory-church/

Leo Provers

- Steen & Benzmüller (2021), Extensional Higher-Order Paramodulation in Leo-III, J. Autom. Reason., http://doi.org/10.1007/s10817-021-09588-x
- Benzmüller, Sultana, Paulson (2015), Thhe Higher-Order Prover Leo-II, J. Autom. Reason., http://doi.org/10.1007/s10817-015-9348-y
- Benzmüller & Kohlhase (1998), LEO – A Higher-Order Theorem Prover, CADE, http://doi.org/10.1007/BFb0054256

LogiKEy & Universal (Meta-)Logical Reasoning

- Benzmüller (2019), **Universal (meta-)logical reasoning: Recent successes.** Sci. Comp. Progr. (http://doi.org/10.1016/j.scico.2018.10.008)
- __, Parent & van der Torre (2020), **Designing normative theories for ethical and legal reasoning: LogiKEy framework, methodology, and tool support.** Artificial Intelligence. (https://doi.org/10.1016/j.artint.2020.103348)
- Benzmüller et.al. (2020), **LogiKEy Workbench: Deontic Logics, Logic Combinations and Expressive Ethical and Legal Reasoning (Isabelle/HOL Dataset)**. Data in Brief (https://doi.org/10.1016/j.dib.2020.106409)
- Bibel (2022), **Computer Kreiert Wissenschaft**. Informatik Spektrum. (https://doi.org/10.1007/s00287-022-01456-1)

Reading (cont'd)

(preprints: http://christoph-benzmueller.de/publications.html)

Foundational Studies in Metaphysics

- Kirchner, Benzmüller & Zalta (2019), **Computer Science and Metaphysics: A Cross-Fertilization**. Open Philosophy (http://doi.org/10.1515/opphil-2019-0015)
- __ (2020), **Mechanizing Principia Logico-Metaphysica in Functional Type Theory**. Review of Symbolic Logic (https://doi.org/10.1017/S1755020319000297)
- Kirchner (2021). **Embedding of Abstract Object Theory in Isabelle/HOL.** Full sources. (https://github.com/ekpyron/AOT/tree/dissertation)
- __ (2022), **Computer-Verified Foundations of Metaphysics and an Ontology of Natural Numbers in Isabelle/HOL**. PhD thesis, Dep. of Maths and CS, FU Berlin.

Foundational Studies in Ethics & Law

- Fuenmayor & Benzmüller (2018), **Formalisation and Evaluation of Alan Gewirth's Proof for the Principle of Generic Consistency in Isabelle/HOL**. (https://www.isa-afp.org/entries/GewirthPGCProof.html)
- __ (2019), **Harnessing Higher-Order (Meta-)Logic to Represent and Reason with Complex Ethical Theories**. PRICAI 2019 (https://doi.org/10.1007/978-3-030-29908-8_34)
- Benzmüller & Fuenmayor (2021), **Value-oriented Legal Argumentation in Isabelle/HOL**. ITP 2021 (https://doi.org/10.4230/LIPIcs.ITP.2021.7)
- __, Fuenmayor & Lomfeld (2022), **Modelling Value-oriented Legal Reasoning in LogiKEy**. (https://arxiv.org/abs/2006.12789)

Studies on the Ontological Argument

- Benzmüller & Woltzenlogel Paleo (2014), **Automating Gödel's Ontological Proof of God's Existence with Higher-order Automated Theorem Provers.** ECAI 2014 (http://doi.org/10.3233/978-1-61499-419-0-93)
- __ (2016), **The Inconsistency in Gödel's Ontological Argument: A Success Story for AI in Metaphysics.** IJCAI 2016 (http://www.ijcai.org/Proceedings/16/Papers/137.pdf)

Reading (cont'd)

(preprints: http://christoph-benzmueller.de/publications.html)

Studies on the Ontological Argument (cont'd)

- Benzmüller & Woltzenlogel Paleo (2016), **An Object-Logic Explanation for the Inconsistency in Gödel's Ontological Theory** (Extended Abstract, Sister Conferences). KI 2016 (http://doi.org/10.1007/978-3-319-46073-4)
- __ (2017). **Computer-Assisted Analysis of the Anderson-Hájek Controversy.** Logica Universalis (http://doi.org/10.1007/s11787-017-0160-9)
- Fuenmayor & Benzmüller (2017), **Automating Emendations of the Ontological Argument in Intensional Higher-Order Modal Logic.** KI 2017 (http://doi.org/10.1007/978-3-319-67190-1_9)
- __ (2018). **A Case Study On Computational Hermeneutics: E. J. Lowe's Modal Ontological Argument.** IfCoLoG Journal of Logics and their Applications (https://www.researchgate.net/publication/333804824)
- Benzmüller & Fuenmayor (2020), **Computer-supported Analysis of Positive Properties, Ultrafilters and Modal Collapse in Variants of Gödel's Ontological Argument.** Bulletin of the Section of Logic (http://doi.org/10.18778/0138-0680.2020.08)
- Benzmüller (2020), **A (Simplified) Supreme Being Necessarily Exists, says the Computer: Computationally Explored Variants of Gödel's Ontological Argument.** KR 2020 (https://doi.org/10.24963/kr.2020/80)
- __ (2022), **A Simplified Variant of Gödel's Ontological Argument.** To appear (https://www.researchgate.net/publication/358607847)

Isabelle/HOL:

- Website: https://isabelle.in.tum.de
- Documentation: https://isabelle.in.tum.de/documentation.html

- Nipkow, Paulson & Wenzel (2002), **Isabelle/HOL: A Proof Assistant for Higher-Order Logic.** Springer (https://doi.org/10.1007/3-540-45949-9)
- Blanchette, Bulwahn & Tobias Nipkow (2011), **Automatic Proof and Disproof in Isabelle/HOL**. FroCoS 2011 (https://doi.org/10.1007/978-3-642-24364-6_2)