

Uniform Proofs via Shallow Semantic Embeddings?

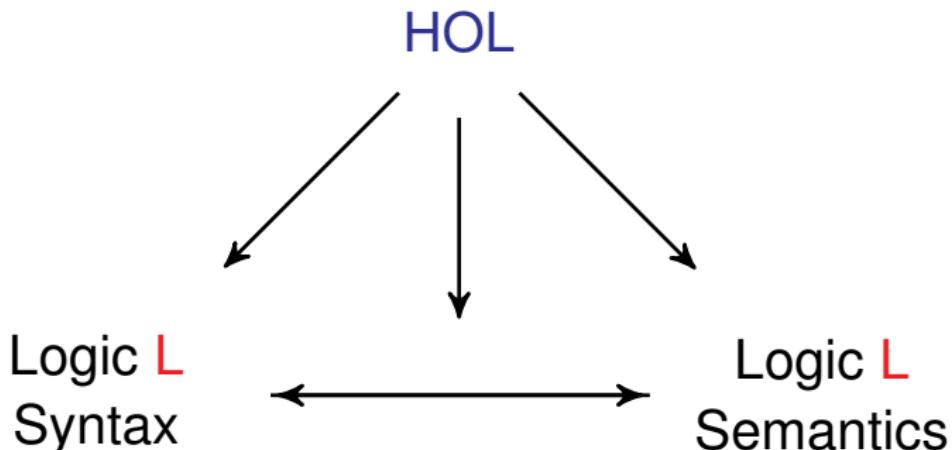
Christoph Benzmüller¹ — FU Berlin

Dagstuhl Seminar on “Universality of Proofs”, October 20, 2016



¹Supported by DFG Heisenberg Fellowship BE 2501/9-1/2

HOL as a (quite) Universal Metalogic via Shallow Semantic Embeddings



Examples for L we have already studied:

Modal Logics, Temporal Logics, Conditional Logics, Intuitionistic Logics, Nominal Logics, Multivalued Logics, (Mathematical) Fuzzy Logics, Access Control Logics, Paraconsistent Logics, Free Logics, Dynamic Logics, ...

Works also for (first-order & higher-order) quantifiers

Application Example A: Computational Metaphysics

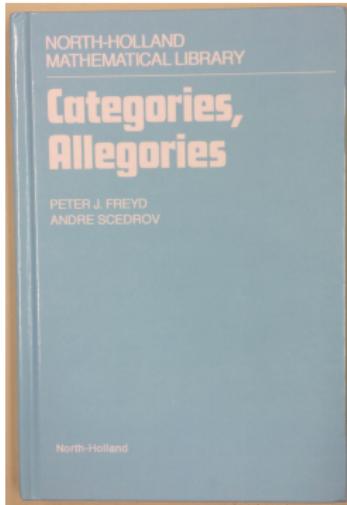


Leibniz: *Calculemus!*



- ▶ Objective: Analysis of Rational Arguments in Metaphysics
- ▶ Joint work with: Bruno Woltzenlogel Paleo (ANU)
- ▶ Main papers: ECAI'2014, IJCAI'2016
- ▶ Logics: **Higher-Order Modal Logics**
 - ▶ modalities: K, KB, S4, S5, ...
 - ▶ quantifiers: constant/varying/increasing/decreasing domains
 - ▶ (restricted comprehension schemes)
- ▶ Results: significant novel findings by ATPs

Application Example B: Theory Exploration — Category Theory



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:

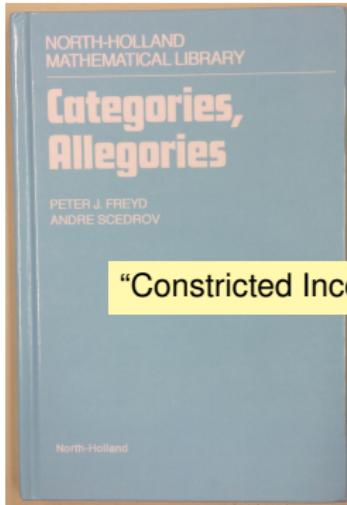
- $\square x$ the source of x ,
- $x\square$ the target of x ,
- xy the composition of x and y .

The axioms:

- P1 xy is defined iff $x\square = \square y$,
- P2a $(\square x)\square = \square x$ and $\square(x\square) = x\square$, P2b
- P3a $(\square x)x = x$ and $x(\square x) = x$, P3b
- P4 $\square(xy) = \square(x\square y)$ and $(xy)\square = ((x\square)y)\square$, P4b
- P5 $x(yz) = (xy)z$.

- ▶ Objective: Exploration and Analysis of Axiom Systems for Category Theory
- ▶ Joint work with: Dana Scott
- ▶ Main papers: ICMS'2016, arXiv'2016
- ▶ Logics: Free Logic
- ▶ Results:
 - ▶ development of 6 related (equivalent) axiom systems for category theory
 - ▶ from Monoids ... via Scott's (1977) axiom system ... to Freyd/Scedrov (1990)
 - ▶ for Freyd and Scedrov (1990) we revealed some flaws/issues

Application Example B: Theory Exploration — Category Theory



“Constricted Inconsistency” or “Missing Axioms/Conditions”

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:

$\square x$ the source of x ,

$x\square$ the target of x ,

the axioms:

A1 xy is defined iff $x\square = \square y$,

A2a $(\square x)\square = \square x$ and $\square(x\square) = x\square$, A2b

A3a $(\square x)x = x$ and $x(x\square) = x$, A3b

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

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

- ▶ Objective: Exploration and Analysis of Axiom Systems for Category Theory
- ▶ Joint work with: Dana Scott
- ▶ Main papers: ICMS'2016, arXiv'2016
- ▶ Logics: Free Logic
- ▶ Results:
 - ▶ development of 6 related (equivalent) axiom systems for category theory
 - ▶ from Monoids ... via Scott's (1977) axiom system ... to Freyd/Scedrov (1990)
 - ▶ for Freyd and Scedrov (1990) we revealed some flaws/issues

Application Example C: Formalisation/Automation of Principia Metaphysica (Zalta)



—since '83→

NOTE: This is an excerpt from an incomplete draft of the monograph *Principia Logico-Metaphysica*. The monograph currently has four parts:

- Part I: Prophilosophy
- Part II: Philosophy
- Part III: Metaphilosophy
- Part IV: Technical Appendices, Bibliography, Index

This excerpt was generated on October 18, 2016 and contains:

- Part II:

Chapter 7: The Language	166
Chapter 8: The Axioms	185
Chapter 9: The Deductive System	203
Chapter 10: Basic Logical Objects	309
Chapter 11: Platonic Forms	354
Chapter 12: Situations, Worlds, and Times	378
Chapter 13: Concepts	438
Chapter 14: Numbers	505

- Part IV:

Appendix: Proofs of Theorems and Metarules	634
Bibliography	849

Principia Logico-Metaphysica

- ▶ Objective: Formalisation/Automation of Principia Metaphysica
- ▶ Joint work with: Daniel Kirchner (student of mine), Ed Zalta (Stanford)
- ▶ Main papers: soon
- ▶ Logics: Hyper-Intensional Higher-Order Modal Logic
- ▶ Results:
 - ▶ first significant part of PM has been formalised (by David)
 - ▶ excellent degree of automation
 - ▶ minor issues detected

DEMO: Computational Metaphysics in Isabelle/HOL

Ontologischer Bereich Feb. 10, 1970

$P(q)$ q is positive ($\Leftrightarrow q \in P$)

Ax 1: $P(p), P(q) \supset P(q \wedge p)$ Ax 2: $P(p) \supset P(\neg p)$

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

P2 $\varphi_{\text{Exis}} \equiv (\forall y)[\varphi(y) \supset N(y)[P(y) \supset \varphi(y)]]$ (Existence)

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

Ax 2 $P(p) \supset N P(p)$ } because it follows
 $\neg P(p) \supset N \neg P(p)$ } from the nature of the
 property

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

Df. $E(x) \equiv \exists p[\varphi_{\text{Exis}} \supset N \neg x \varphi(x)]$ necessary Existence

Ax 3 $P(E)$

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

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

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

" $\supset N(\exists y) G(y)$

any two instances of x are nec. equivalent

exclusive or and for any number of them

$M(x) G(x)$ means all pos. prop. w.r.t. com-
 patible This is true because of:
Ax 4: $P(q), q \supset \neg q \vdash P(\neg q)$ which impl.
 $\begin{cases} x=x & \text{is positive} \\ x \neq x & \text{is negative} \end{cases}$

But if a system S of pos. prop. were incons.
 It would mean, that the non-prop. S (which
 is positive) would be $x \neq x$

Positive means positive in the moral aesthe-
 sical sense (independently of the accidental structure of
 the world). Only \neg in the ax. frame. It is
 also meant "Attribution" as opposed to "Platification"
 (or containing negation). This interprets the pos. prop.

$\exists x \varphi \text{ (possibility)} : (x) N \neg \varphi(x)$ Otherwise: $\varphi(x) \supset x \neq x$
 hence $x \neq x$ positive not $x=x$ i.e. negation. At
 the end of proof Atm

ax. i.e. the normal form in terms of elem. prop. contains
 Member without negation.

DEMO: Computational Metaphysics in Isabelle/HOL

Axiom A1 Either a property or its negation is positive, but not both: $\forall\phi[P(\neg\phi) \leftrightarrow \neg P(\phi)]$

Axiom A2 A property necessarily implied by a positive property is positive:

$$\forall\phi\forall\psi[(P(\phi) \wedge \Box\forall x[\phi(x) \rightarrow \psi(x)]) \rightarrow P(\psi)]$$

Thm. T1 Positive properties are possibly exemplified: $\forall\phi[P(\phi) \rightarrow \Diamond\exists x\phi(x)]$

Def. D1 A *God-like* being possesses all positive properties: $G(x) \leftrightarrow \forall\phi[P(\phi) \rightarrow \phi(x)]$

Axiom A3 The property of being God-like is positive: $P(G)$

Cor. C Possibly, God exists: $\Diamond\exists xG(x)$

Axiom A4 Positive properties are necessarily positive: $\forall\phi[P(\phi) \rightarrow \Box P(\phi)]$

Def. D2 An *essence* of an individual is a property possessed by it and necessarily implying any of its properties: $\phi \text{ ess. } x \leftrightarrow \phi(x) \wedge \forall\psi(\psi(x) \rightarrow \Box\forall y(\phi(y) \rightarrow \psi(y)))$

Thm. T2 Being God-like is an essence of any God-like being: $\forall x[G(x) \rightarrow G \text{ ess. } x]$

Def. D3 Necessary existence of an individual is the necessary exemplification of all its essences: $NE(x) \leftrightarrow \forall\phi[\phi \text{ ess. } x \rightarrow \Box\exists y\phi(y)]$

Axiom A5 Necessary existence is a positive property: $P(NE)$

Thm. T3 Necessarily, God exists: $\Box\exists xG(x)$

DEMO:

The screenshot shows the Isabelle/HOL proof assistant interface. The main window displays the theory file `GodProof.thy`. The code includes several definitions and theorems related to God-like essences and necessary existence. A specific theorem, T3, is highlighted with a yellow background, indicating it is currently being worked on or is the current goal.

```
GodProof.thy (~/chris/trunk/tex/talks/2016-Dagstuhl/)

115 definition ess (infixr "ess" 85) where
116   "Φ ess x = Φ(x) ∧ (∀Φ. Φ(x) → □(∀y. Φ(y) → Ψ(y)))"
117
118 (* T2: Being God-like is an essence of any God-like being *)
119 theorem T2: "[∀x. G(x) → G ess x]" by (smt A1b A4 G_def ess_def)
120
121 (* NE: Necessary existence of an individual is the necessary
122    exemplification of all its essences *)
123 definition NE where "NE(x) = (∀Φ. Φ ess x → □(∃x. Φ(x)))"
124
125 (* A5: Necessary existence is a positive property *)
126 axiomatization where A5: "[P(NE)]"
127
128 (* T3: Necessarily, God exists *)
129 theorem T3: "[□(∃x. G(x))]" by (metis A5 C T2 G_def NE_def S5)
130
131
132 (* Consistency is confirmed by Nitpick *)
133 lemma True nitpick [satisfy, user_axioms] oops
134
```

The proof state at the bottom shows a single subgoal:

```
proof (prove)
goal (1 subgoal):
  1. [| (λw. [S5 w → mexi_prop G]) |]
```

The interface includes a toolbar with various icons, a vertical sidebar with tabs for Documentation, Sidekick, State, and Theories, and a status bar at the bottom showing the file name, memory usage, and timestamp.

DEMO:

The screenshot shows the Isabelle IDE interface with a theory file named `GodProof.thy`. The code defines types for possible worlds, individuals, and a function σ . It includes abbreviations for modal logic connectives (`mneg`, `mand`, `mor`, `mimp`, `mequ`, `mnegpred`) and generic operators (`mboxgen`, `mdiagen`). A cursor is visible at line 20. Below the main editor, a smaller window displays a constant definition `mdiagen`.

```
theory GodProof imports Main
begin
declare [[smt_solver = cvc4]]
typedecl i -- "type for possible worlds"
typedecl μ -- "type for individuals"
type_synonym σ = "(i⇒bool)"

(* Shallow embedding modal logic connectives in HOL *)
abbreviation mneg ("¬_"[52]53) where "¬φ ≡ λw. ¬φ(w)"
abbreviation mand (infixr "∧" 51) where "φ ∧ ψ ≡ λw. φ(w) ∧ ψ(w)"
abbreviation mor (infixr "∨" 50) where "φ ∨ ψ ≡ λw. φ(w) ∨ ψ(w)"
abbreviation mimp (infixr "→" 49) where "φ → ψ ≡ λw. φ(w) → ψ(w)"
abbreviation mequ (infixr "↔" 48) where "φ ↔ ψ ≡ λw. φ(w) ←→ ψ(w)"
abbreviation mnegpred ("¬_"[52]53) where "¬Φ ≡ λx. λw. ¬Φ(x)(w)"

(* Generic box and diamonnd operators *)
abbreviation mboxgen ("□") where "□r φ ≡ λw. ∀v. r w v → φ(v)"
abbreviation mdiagen ("◇") where "◇r φ ≡ λw. ∃v. r w v ∧ φ(v)"

consts
  mdiagen :: "('a ⇒ 'b ⇒ bool) ⇒ ('b ⇒ bool) ⇒ 'a ⇒ bool"
```

DEMO:

The screenshot shows the Isabelle IDE interface with the file `GodProof.thy` open. The code defines various abbreviations and constants for shallow embeddings of quantifiers in HOL. The interface includes toolbars at the top, a vertical navigation bar on the right, and a bottom toolbar with tabs like Output, Query, Sledgehammer, and Symbols.

```
21 (* Shallow embedding of constant domain quantifiers in HOL *)
22 abbreviation mall_const ("!c") where "!c !Phi ≡ !w. !x. !Phi(x)(w)"
23 abbreviation mallB_const (binder "!c"[8]9)
24   where "?c x. !Phi(x) ≡ !c !Phi"
25 abbreviation mexi_const ("?c") where "?c !Phi ≡ !w. ?x. !Phi(x)(w)"
26 abbreviation mexiB_const (binder "?c"[8]9)
27   where "?c x. !Phi(x) ≡ ?c !Phi"
28
29 (* Shallow embedding of varying domain quantifiers in HOL *)
30 consts eiw :: "'a ⇒ 'b ⇒ bool"
31 axiomatization where nonempty: "?w. ?x. eiw w x"
32 abbreviation mall_var ("!v")
33   where "?v !Phi ≡ !w. !x. eiw(w)(x) ⟹ !Phi(x)(w)"
34 abbreviation mallB_var (binder "?v"[8]9)
35   where "?v x. !Phi(x) ≡ ?v !Phi"
36 abbreviation mexi_var ("?v")
37   where "?v !Phi ≡ !w. ?x. eiw(w)(x) ∧ !Phi(x)(w)"
38 abbreviation mexiB_var (binder "?v"[8]9)
39   where "?v x. !Phi(x) ≡ ?v !Phi"
40
```

Bottom status bar: 8,1 (184/4984) (isabelle,isabelle,UTF-8-Isabelle) N m r o UG 200/783MB 11:20 AM

DEMO:

The screenshot shows the Isabelle proof assistant interface with the file `GodProof.thy` open. The code defines various modal logic concepts and accessibility relations.

```
41 (* Validity of a (higher-order) modal logic formula means truth
42   in all possible worlds *)
43 abbreviation mvalid :: "σ ⇒ bool" ("[_]"[7]8)
44   where "[p] ≡ ∀w. p w"
45
46 (* Reflexivity, symmetry and transitivity of relations *)
47 abbreviation ref
48   where "ref r ≡ ∀x. r x x"
49 abbreviation sym
50   where "sym r ≡ ∀x y. r x y → r y x"
51 abbreviation trans
52   where "trans r ≡ ∀x y z. r x y ∧ r y z → r x z"
53
54 (* Different accessibility relations for different modal logics *)
55 consts
56   K :: "i⇒i⇒bool"
57   KB :: "i⇒i⇒bool"
58   S4 :: "i⇒i⇒bool"
59   S5 :: "i⇒i⇒bool"
60 axiomatization where
61   KB: "sym KB" and
62   S4: "ref S4 ∧ trans S4" and
63   S5: "ref S5 ∧ sym S5 ∧ trans S5"
64
```

At the bottom, there are checkboxes for "Proof state" and "Auto update", an "Update" button, a search bar, and a zoom level of 100%. The menu bar includes "Output", "Query", "Sledgehammer", and "Symbols". The status bar at the bottom shows "30,36 (1260/4984)" and "Isabelle parsing complete... (isabelle,isabelle,UTF-8-Isabelle) N m r o UG 50/772MB 11:21 AM".

DEMO:

The screenshot shows the Isabelle IDE interface with the file `GodProof.thy` open. The code is written in ML-like syntax with some binder annotations. The interface includes toolbars at the top, a vertical navigation bar on the right, and various status indicators at the bottom.

```
GodProof.thy
(* **** *)
(* Setting Parameters for Experimentation with Ontological Argument *)
(* Default: Second-Order S5 with constant domain quantifiers *)
abbreviation mbox ("□")
where "□ ≡ □S5"
abbreviation mdia ("◊")
where "◊ ≡ ◊S5"
abbreviation mall_ind :: "(μ ⇒ σ) ⇒ σ" (binder "∀i" [8] 9)
where "∀x. φ(x) ≡ ∀c φ"
abbreviation mexi_ind :: "(μ ⇒ σ) ⇒ σ" (binder "∃i" [8] 9)
where "∃x. φ(x) ≡ ∃c φ"
abbreviation mall_prop :: "((μ ⇒ σ) ⇒ σ) ⇒ σ" (binder "∀p" [8] 9)
where "∀px. φ(x) ≡ ∀c φ"
abbreviation mexi_prop :: "((μ ⇒ σ) ⇒ σ) ⇒ σ" (binder "∃p" [8] 9)
where "∃px. φ(x) ≡ ∃c φ"
(* **** *)
(* Analysis of Gödel's Ontological Argument (in Scott's version) *)
```

consts
trans :: "('a ⇒ 'a ⇒ bool) ⇒ bool"

Proof state Auto update Update Search: 100%
Output Query Sledgehammer Symbols
51,21 (1966/4984) (isabelle,isabelle,UTF-8-Isabelle) Nm ro UG 416/772MB 11:22 AM

DEMO:

The screenshot shows the Isabelle IDE interface with the file `GodProof.thy` open. The code implements Gödel's Ontological Argument in Scott's version. It includes axioms A1, A2, and A3, and theorems T1 and God. The interface features a toolbar at the top, a vertical navigation bar on the right, and various status indicators at the bottom.

```
85 (* ****  
86 (* Analysis of Gödel's Ontological Argument (in Scott's version) *)  
87  
88 (* Positiveness *)  
89 consts P :: " $(\mu \Rightarrow \sigma) \Rightarrow \sigma$ "  
90  
91 axiomatization where  
92 (* A1: Either a property or its negation is positive, but not both *)  
93   Ala: " $[\forall \Phi. P(\neg \Phi) \rightarrow \neg P(\Phi)]$ " and  
94   Alb: " $[\forall \Phi. \neg P(\Phi) \rightarrow P(\neg \Phi)]$ " and  
95 (* A2: A property necessarily implied by a positive property is positive *)  
96   A2: " $[\forall \Phi. \Psi. P(\Phi) \wedge \Box(\forall x. \Phi(x) \rightarrow \Psi(x)) \rightarrow P(\Psi)]$ "  
97  
98 (* T1: Positive properties are possibly exemplified *)  
99 theorem T1: " $[\forall \Phi. P(\Phi) \rightarrow \Diamond(\exists x. \Phi(x))]$ " using Ala A2 by blast  
100  
101 (* God: A God-like being possesses all positive properties *)  
102 definition G where "G(x) = ( $\forall \Phi. P(\Phi) \rightarrow \Phi(x)$ )"  
103  
104 (* A3: The property of being God-like is positive *)  
105 axiomatization where A3: " $[P(G)]$ "  
106
```

At the bottom, there are checkboxes for "Proof state" and "Auto update", an "Update" button, a "Search:" field, and a zoom level of "100%". The status bar at the very bottom shows the file path, memory usage (105/758MB), and the current time (11:22 AM).

consts
 mdia :: " $(i \Rightarrow \text{bool}) \Rightarrow i \Rightarrow \text{bool}$ "

Output | Query | Sledgehammer | Symbols

72,28 (2572/4984) (isabelle,isabelle,UTF-8-Isabelle) Nm r o UG 105/758MB 11:22 AM

DEMO:

GodProof.thy (modified)

```
85 (* **** Analysis of Gödel's Ontological Argument (in Scott's version) **** *)
86 (* Analysis of Gödel's Ontological Argument (in Scott's version) *)
87
88 (* Positiveness *)
89 consts P :: " $(\mu \Rightarrow \sigma) \Rightarrow \sigma$ "
90
91 axiomatization where
92 (* A1: Either a property or its negation is positive, but not both *)
93   Ala: " $[\forall \Phi. P(\Phi) \rightarrow \neg P(\Phi)]$ " and
94   Alb: " $[\forall \Phi. \neg P(\Phi) \rightarrow P(\neg \Phi)]$ " and
95 (* A2: A property necessarily implied by a positive property is positive *)
96   A2: " $[\forall \Phi \Psi. P(\Phi) \wedge \Box(\forall x. \Phi(x) \rightarrow \Psi(x)) \rightarrow P(\Psi)]$ "
97
98 (* T1: Positive properties are possibly exemplified *)
99 theorem T1: " $[\forall \Phi. P(\Phi) \rightarrow \Diamond(\exists x. \Phi(x))]$ " sledgehammer
100
101 (* God: A God-like being possesses all positive properties *)
102 definition G where "G(x) =  $(\forall \Phi. P(\Phi) \rightarrow \Phi(x))$ "
103
104 (* A3: The property of being God-like is positive *)
105 axiomatization where A3: " $[P(G)]$ "
106
```

Sledgehammering...

"e": Try this: using Ala A2 by blast (43 ms).
"spass": Try this: using Ala A2 by blast (49 ms).
"z3": Try this: using Ala A2 by blast (162 ms).

Output Query Sledgehammer Symbols

99,61 (3569/4975) Input/output complete (isabelle,isabelle,UTF-8–Isabelle) N m r o UG 265/758MB 11:23 AM

DEMO:

The screenshot shows the Isabelle IDE interface with the file `GodProof.thy` open. The code is a浅层语义嵌入 (Shallow Semantic Embedding) of Gödel's Ontological Argument.

```
85>(* **** * Analysis of Gödel's Ontological Argument (in Scott's version) *)
86 consts P :: "(μ⇒σ)⇒σ"
87
88(* Positiveness *)
89
90 axiomatization where
91 (* A1: Either a property or its negation is positive, but not both *)
92   Ala: "[∀Φ. P(¬Φ) → ¬P(Φ)]" and
93   Alb: "[∀Φ. ¬P(Φ) → P(¬Φ)]" and
94 (* A2: A property necessarily implied by a positive property is positive *)
95   A2: "[∀Φ Ψ. P(Φ) ∧ □(∀x. Φ(x) → Ψ(x)) → P(Ψ)]"
96
97 (* T1: Positive properties are possibly exemplified *)
98 theorem T1: "[∀Φ. P(Φ) → ◇(∃x. Φ(x))]" using Ala A2 by blast
99
100 (* God: A God-like being possesses all positive properties *)
101 definition G where "G(x) = ( ∀Φ. P(Φ) → Φ(x)) "
102
103 (* A3: The property of being God-like is positive *)
104 axiomatization where A3: "[P(G)]"
105
106
```

The proof state at the bottom shows the goal being proved:

```
proof (prove)
using this:
  • [ (λw. ∀x. (P (¬x) → (¬P) x) w) ]
  • [ (λw. ∀x xa. P x w ∧ ([ (λv. S5 w v → ( ∀xb. (x xb → xa xb) v)) ] →
    P xa w) ) ]
```

Toolbars and status bar details:

- Toolbar icons: File, Open, Save, Print, Undo, Redo, Cut, Copy, Paste, Find, Replace, etc.
- Documentation, Sidekick, State, Theories buttons on the right.
- Status bar: 99,49 (3557/4984) Isabelle parsing complete... (isabelle,isabelle,UTF-8-Isabelle) N m r o UG 358/758MB 11:24 AM

DEMO:

The screenshot shows the Isabelle/HOL proof assistant interface. The main window displays the theory file `GodProof.thy` with the following content:

```
101 (* God: A God-like being possesses all positive properties *)
102 definition G where "G(x) = (Vp. P(Φ) → Φ(x))"
103
104 (* A3: The property of being God-like is positive *)
105 axiomatization where A3: "[P(G)]"
106
107 (* C: Possibly, God exists *)
108 corollary C: "[◊(Ex. G(x))]" by (metis A3 T1)
109
110 (* A4: Positive properties are necessarily positive *)
111 axiomatization where A4: "[Vp. P(Φ) → □(P(Φ))]"
112
113 (* Ess: An essence of an individual is a property possessed by
114     it and necessarily implying any of its properties: *)
115 definition ess (infixr "ess" 85) where
116     "Φ ess x = Φ(x) ∧ (Vp. Φ(x) → □(Vy. Φ(y) → Φ(y)))"
117
118 (* T2: Being God-like is an essence of any God-like being *)
119 theorem T2: "[Vx. G(x) → G ess x]" by (smt Alb A4 G_def ess_def)
120
121 (* NE: Necessary existence of an individual is the necessary
122     exemplification of all its essences *)
```

The cursor is positioned over the theorem `T2`. Below the main text area, there is a toolbar with checkboxes for "Proof state" and "Auto update", an "Update" button, a "Search:" field, and a zoom level of "100%".

At the bottom, there is a menu bar with "Output", "Query", "Sledgehammer", and "Symbols". The status bar at the bottom right shows the file name `(isabelle,isabelle,UTF-8-Isabelle)`, the number of goals `119,70 (4342/4984)`, memory usage `467/758MB`, and the current time `11:25 AM`.

DEMO:

The screenshot shows the Isabelle proof assistant interface with the file `GodProof.thy` open. The code is a modal logic proof involving concepts like God-like existence, necessary existence, and consistency.

```
GodProof.thy (~/chris/trunk/tex/talks/2016-Dagstuhl/)

116 "Φ ess x = Φ(x) ∧ ( ∀ψ. ψ(x) → □( ∀y. Φ(y) → ψ(y)))"
117
118 (* T2: Being God-like is an essence of any God-like being *)
119 theorem T2: "[ ∀x. G(x) → G ess x]" by (smt A1b A4 G_def ess_def)
120
121 (* NE: Necessary existence of an individual is the necessary
122   exemplification of all its essences *)
123 definition NE where "NE(x) = ( ∀φ. Φ ess x → □( ∃x. Φ(x)))"
124
125 (* A5: Necessary existence is a positive property *)
126 axiomatization where A5: "[P(NE)]"
127
128 (* T3: Necessarily, God exists *)
129 theorem T3: "[□( ∃x. G(x))]" by (metis A5 C T2 G_def NE_def S5)
130
131 (* Consistency is confirmed by Nitpick *)
132 lemma True nitpick [satisfy, user_axioms] oops
133
134
135 (*
136 lemma MC: "[ ∀φ. φ → ( □ φ)]" -- {* Modal Collapse *}
137 theorem T3: [ (λw. [S5 w → mexi_prop G])]
```

The interface includes a toolbar at the top, a vertical navigation bar on the right with tabs for Documentation, Sidekick, State, and Theories, and a status bar at the bottom showing file information and memory usage.

DEMO:

```
GodProof.thy (~/chris/trunk/tex/talks/2016-Dagstuhl/)

65>(* **** *)
66
67(* Setting Parameters for Experimentation with Ontological Argument *)
68(* Default: Second-Order S5 with constant domain quantifiers *)
69
70abbreviation mbox ("□")
71  where "□ ≡ □S5"
72abbreviation mdia ("◇")
73  where "◇ ≡ ◇S5"
74
75abbreviation mall_ind :: "(μ ⇒ σ) ⇒ σ" (binder"∀¹[8]9")
76  where "∀¹x. φ(x) ≡ ∀c φ"
77abbreviation mexi_ind :: "(μ ⇒ σ) ⇒ σ" (binder"∃¹[8]9")
78  where "∃¹x. φ(x) ≡ ∃c φ"
79
80abbreviation mall_prop :: "((μ ⇒ σ) ⇒ σ) ⇒ σ" (binder"∀º[8]9")
81  where "∀ºx. φ(x) ≡ ∀c φ"
82abbreviation mexi_prop :: "(μ ⇒ σ) ⇒ σ" (binder"∃º[8]9")
83  where "∃ºx. φ(x) ≡ ∃c φ"
84
85(* **** *)
86(* Analysis of Gödel's Ontological Argument (in Scott's version) *)
```

consts
mbox :: "(i ⇒ bool) ⇒ i ⇒ bool"

Output Query Sledgehammer Symbols

71,19 (2544/4980) (isabelle,isabelle,UTF-8-Isabelle) N m r o UG 147/723MB 11:28 AM

DEMO:

The screenshot shows the Isabelle IDE interface with the file `GodProof.thy` open. The code is written in ML-like syntax with some annotations in red. The interface includes a toolbar at the top, a navigation bar on the right, and a status bar at the bottom.

```
65 (* **** *)
66
67 (* Setting Parameters for Experimentation with Ontological Argument *)
68 (* Default: Second-Order S5 with constant domain quantifiers *)
69
70 abbreviation mbox ("□")
71   where "□ ≡ □KB"
72 abbreviation mdia ("◇")
73   where "◇ ≡ ◇KB"
74
75 abbreviation mall_ind :: "(μ ⇒ σ) ⇒ σ" (binder "∀1" [8]9)
76   where "∀1x. φ(x) ≡ ∀c φ"
77 abbreviation mexi_ind :: "(μ ⇒ σ) ⇒ σ" (binder "∃1" [8]9)
78   where "∃1x. φ(x) ≡ ∃c φ"
79
80 abbreviation mall_prop :: "((μ ⇒ σ) ⇒ σ) ⇒ σ" (binder "∀0" [8]9)
81   where "∀0x. φ(x) ≡ ∀c φ"
82 abbreviation mexi_prop :: "((μ ⇒ σ) ⇒ σ) ⇒ σ" (binder "∃0" [8]9)
83   where "∃0x. φ(x) ≡ ∃c φ"
84
85 (* **** *)
86 (* Analysis of Gödel's Ontological Argument (in Scott's version *)
```

consts
mdia :: "(i ⇒ bool) ⇒ i ⇒ bool"

✓ Proof state ✓ Auto update Update Search: 100%

Output Query Sledgehammer Symbols

73,20 (2592/4980) Isabelle parsing complete... (isabelle,isabelle,UTF-8-Isabelle) Nm ro UG 339/723MB 11:29 AM

DEMO:

```
GodProof.thy (modified)
GodProof.thy (~/chris/trunk/tex/talks/2016-Dagstuhl/)

(* **** *)
(* Setting Parameters for Experimentation with Ontological Argument *)
(* Default: Second-Order S5 with constant domain quantifiers *)
abbreviation mbox ("□")
  where "□" ≡ □KB
abbreviation mdia ("◇")
  where "◇" ≡ ◇KB
abbreviation mall_ind :: "(μ⇒σ)⇒σ" (binder"∀¹"[8]9)
  where "∀x. φ(x) ≡ ∀v φ"
abbreviation mexi_ind :: "(μ⇒σ)⇒σ" (binder"∃¹"[8]9)
  where "∃x. φ(x) ≡ ∃v φ"
abbreviation mall_prop :: "((μ⇒σ)⇒σ)⇒σ" (binder"∀²"[8]9)
  where "∀x. φ(x) ≡ ∀c φ"
abbreviation mexi_prop :: "((μ⇒σ)⇒σ)⇒σ" (binder"∃²"[8]9)
  where "∃x. φ(x) ≡ ∃c φ"
(* **** *)
(* Analysis of Gödel's Ontological Argument (in Scott's version) *)

consts
  mexi_ind :: "(μ ⇒ i ⇒ bool) ⇒ i ⇒ bool"
```

✓ Proof state ✓ Auto update Update Search: 100%
consts
mexi_ind :: "(μ ⇒ i ⇒ bool) ⇒ i ⇒ bool"

Output Query Sledgehammer Symbols
78,33 (2771/4980) Input/output complete (isabelle,isabelle,UTF-8-Isabelle)Nr nro UG 420/723MB 11:30 AM

DEMO:

The screenshot shows the Isabelle IDE interface with the file `GodProof.thy` open. The code is a proof script for the `God` example, structured as follows:

```
91 axiomatization where
92 (* A1: Either a property or its negation is positive, but not both *)
93   Ala: " $\exists \Phi. P(\Phi) \rightarrow \neg P(\Phi)$ " and
94   Alb: " $\exists \Phi. \neg P(\Phi) \rightarrow P(\neg\Phi)$ " and
95 (* A2: A property necessarily implied by a positive property is positive *)
96   A2: " $\forall \Phi \Psi. P(\Phi) \wedge \Box(\forall x. \Phi(x) \rightarrow \Psi(x)) \rightarrow P(\Psi)$ "
97
98 (* T1: Positive properties are possibly exemplified *)
99 theorem T1: " $\exists \Phi. P(\Phi) \rightarrow \Diamond(\exists x. \Phi(x))$ " using Ala A2 by blast
100
101 (* God: A God-like being possesses all positive properties *)
102 definition G where "G(x) = ( $\forall \Phi. P(\Phi) \rightarrow \Phi(x)$ )"
103
104 (* A3: The property of being God-like is positive *)
105 axiomatization where A3: " $\exists \Phi. P(G)$ "
106
107 (* C: Possibly, God exists *)
108 corollary C: " $\Diamond(\exists x. G(x))$ " by (metis A3 T1)
109
110 (* A4: Positive properties are necessarily positive *)
111 axiomatization where A4: " $\forall \Phi. P(\Phi) \rightarrow \Box(P(\Phi))$ "
112
```

The proof state at the bottom of the editor window is:

```
theorem T1:  $\lambda w. \forall x. P x w \longrightarrow (\exists v. (KB w \wedge \text{mexi\_ind } x \ v))$ 
```

Below the editor window, the status bar displays:

```
99,70 (3578/4980) (isabelle,isabelle,UTF-8-Isabelle)N m r o UG 246/712MB 11:30 AM
```

DEMO:

The screenshot shows the Isabelle proof assistant interface with the file `GodProof.thy` open. The code defines several properties and proves the existence of a God-like being.

```
114      it and necessarily implying any of its properties: *)  
115 definition ess (infixr "ess" 85) where  
116   " $\Phi \text{ ess } x = \Phi(x) \wedge (\forall \Psi. \Psi(x) \rightarrow \square(\forall y. \Phi(y) \rightarrow \Psi(y)))$ "  
117  
118 (* T2: Being God-like is an essence of any God-like being *)  
119 theorem T2: " $\forall x. G(x) \rightarrow G \text{ ess } x$ " by (smt Alb A4 G_def ess_def)  
120  
121 (* NE: Necessary existence of an individual is the necessary  
122   exemplification of all its essences *)  
123 definition NE where " $NE(x) = (\forall \Phi. \Phi \text{ ess } x \rightarrow \square(\exists x. \Phi(x)))$ "  
124  
125 (* A5: Necessary existence is a positive property *)  
126 axiomatization where A5: "[P(NE)]"  
127  
128 (* T3: Necessarily, God exists *)  
129 theorem T3: " $\square(\exists x. G(x))$ " by (metis A5 C T2 G_def NE_def S5)  
130  
131  
132 (* Consistency is confirmed by Nitpick *)  
133 lemma True nitpick [satisfy, user_axioms] oops  
134  
135
```

The interface includes a toolbar at the top, a vertical sidebar on the right with tabs for Documentation, Sidekick, State, and Theories, and a bottom navigation bar with tabs for Output, Query, Sledgehammer, and Symbols. The status bar at the bottom shows file statistics: 129,67 (4711/4980), (isabelle,isabelle,UTF-8-Isabelle)N m r o UG 336/712MB 11:31 AM.

DEMO:

The screenshot shows the Isabelle proof assistant interface with the file `GodProof.thy` open. The code defines several properties and theorems related to God-like existence and necessary existence.

```
114      it and necessarily implying any of its properties: *)
115 definition ess (infixr "ess" 85) where
116   " $\Phi \text{ ess } x = \Phi(x) \wedge (\forall \Psi. \Psi(x) \rightarrow \square(\forall y. \Phi(y) \rightarrow \Psi(y)))$ "
117
118 (* T2: Being God-like is an essence of any God-like being *)
119 theorem T2: " $[\forall^1 x. G(x) \rightarrow G \text{ ess } x]$ " by (smt Alb A4 G_def ess_def)
120
121 (* NE: Necessary existence of an individual is the necessary
122    exemplification of all its essences *)
123 definition NE where "NE(x) = ( $\forall \Phi. \Phi \text{ ess } x \rightarrow \square(\exists^1 x. \Phi(x))$ )"
124
125 (* A5: Necessary existence is a positive property *)
126 axiomatization where A5: "[P(NE)]"
127
128 (* T3: Necessarily, God exists *)
129 theorem T3: " $[\square(\exists^1 x. G(x))]$ " by (metis A5 C T2 G_def NE_def KB)
130
131
132 (* Consistency is confirmed by Nitpick *)
133 lemma True nitpick [satisfy, user_axioms] oops
134
135
```

The theorem `T3` is highlighted in yellow, indicating it is the current goal. The status bar at the bottom shows the command `theorem T3: [(\lambda w. [KB w → mexi_ind G])]`.

Bottom navigation bar: Output, Query, Sledgehammer, Symbols

Bottom status bar: 129,67 (4711/4980) (isabelle,isabelle,UTF-8-Isabelle) UG 418/72.2MB 11:32 AM

DEMO:

The screenshot shows the Isabelle IDE interface with the file `GodProof.thy` open. The code is written in ML-like syntax with some binder annotations. The interface includes a toolbar at the top, a vertical navigation bar on the right, and various status indicators at the bottom.

```
65 (* **** *)
66
67 (* Setting Parameters for Experimentation with Ontological Argument *)
68 (* Default: Second-Order S5 with constant domain quantifiers *)
69
70 abbreviation mbox ("□")
71   where "□ ≡ □KB"
72 abbreviation mdia ("◇")
73   where "◇ ≡ ◇KB"
74
75 abbreviation mall_ind :: "(μ ⇒ σ) ⇒ σ" (binder "∀¹" [8] 9)
76   where "∀¹ x. φ(x) ≡ ∀v φ"
77 abbreviation mexi_ind :: "(μ ⇒ σ) ⇒ σ" (binder "∃¹" [8] 9)
78   where "∃¹ x. φ(x) ≡ ∃v φ"
79
80 abbreviation mall_prop :: "((μ ⇒ σ) ⇒ σ) ⇒ σ" (binder "∀²" [8] 9)
81   where "∀² x. φ(x) ≡ ∀c φ"
82 abbreviation mexi_prop :: "((μ ⇒ σ) ⇒ σ) ⇒ σ" (binder "∃²" [8] 9)
83   where "∃² x. φ(x) ≡ ∃c φ"
84
85 (* **** *)
86 (* Analysis of Gödel's Ontological Argument (in Scott's version) *)
```

At the bottom of the code editor, there are checkboxes for "Proof state" and "Auto update", a "Update" button, a "Search" field, and a zoom level indicator set to 100%. Below the editor, a "consts" section defines the type of the `mbox` abbreviation.

```
consts
  mbox :: "(i ⇒ bool) ⇒ i ⇒ bool"
```

The status bar at the bottom shows the file number (70,28), the total number of files (2525/4980), the encoding (UTF-8), the memory usage (UG 215/703MB), and the current time (11:33 AM).

DEMO:

The screenshot shows the Isabelle IDE interface with the file `GodProof.thy` open. The code defines various abbreviations and constants for modal logic and Gödel's Ontological Argument.

```
65 (* ****  
66  
67 (* Setting Parameters for Experimentation with Ontological Argument *)  
68 (* Default: Second-Order S5 with constant domain quantifiers *)  
69  
70 abbreviation mbox ("□")  
71   where "□ ≡ □K"  
72 abbreviation mdia ("◇")  
73   where "◇ ≡ ◇K"  
74  
75 abbreviation mall_ind :: "(μ ⇒ σ) ⇒ σ" (binder "∀¹" [8] 9)  
76   where "∀¹ x. φ(x) ≡ ∀c φ"  
77 abbreviation mexi_ind :: "(μ ⇒ σ) ⇒ σ" (binder "∃¹" [8] 9)  
78   where "∃¹ x. φ(x) ≡ ∃c φ"  
79  
80 abbreviation mall_prop :: "((μ ⇒ σ) ⇒ σ) ⇒ σ" (binder "∀º" [8] 9)  
81   where "∀º x. φ(x) ≡ ∀c φ"  
82 abbreviation mexi_prop :: "((μ ⇒ σ) ⇒ σ) ⇒ σ" (binder "∃º" [8] 9)  
83   where "∃º x. φ(x) ≡ ∃c φ"  
84  
85 (* ****  
86 (* Analysis of Gödel's Ontological Argument (in Scott's version) *)
```

At the bottom, there is a search bar with checkboxes for "Proof state" and "Auto update", an "Update" button, and a "Search:" field. Below the search bar, the word "consts" is followed by the definition of `mexi_ind`.

Bottom status bar: 79,1 (2770/4978) Isabelle parsing complete... (isabelle,isabelle,UTF-8-Isabelle) Nm n o UG 385/703MB 11:34 AM

DEMO:

The screenshot shows the Isabelle proof assistant interface with the file `GodProof.thy` open. The code defines several concepts and proves theorems related to God-like essences and necessary existence.

```
114      it and necessarily implying any of its properties: *)
115 definition ess (infixr "ess" 85) where
116   "Φ ess x = Φ(x) ∧ (∀Ψ. Ψ(x) → □(∀y. Φ(y) → Ψ(y)))"
117
118 (* T2: Being God-like is an essence of any God-like being *)
119 theorem T2: "[∀x. G(x) → G ess x]" by (smt Alb A4 G_def ess_def)
120
121 (* NE: Necessary existence of an individual is the necessary
122    exemplification of all its essences *)
123 definition NE where "NE(x) = (∀Φ. Φ ess x → □(∃x. Φ(x)))"
124
125 (* A5: Necessary existence is a positive property *)
126 axiomatization where A5: "[P(NE)]"
127
128 (* T3: Necessarily, God exists *)
129 theorem T3: "[□(∃x. G(x))]" by (metis A5 C T2 G_def NE_def)
130
131
132 (* Consistency is confirmed by Nitpick *)
133 lemma True nitpick [satisfy, user_axioms] oops
134
```

The theorem `T3` is highlighted with a yellow background. At the bottom of the interface, there is a search bar and a status bar indicating the proof state, auto update, and memory usage.

Bottom navigation bar:

- Output
- Query
- Sledgehammer
- Symbols

Status bar:

- 129,34 (4676/4975)
- (isabelle,isabelle,UTF-8-Isabelle)N m r o UG
- 258/656MB 11:39 AM

DEMO:

The screenshot shows the Isabelle IDE interface with the file `GodProof.thy` open. The code is as follows:

```
124
125 (* A5: Necessary existence is a positive property *)
126 axiomatization where A5: "[P(NE)]"
127
128 (* T3: Necessarily, God exists *)
129 theorem T3: "[□(∃ix. G(x))]"
130 nitpick [show_all, format=]
131 by (metis A5 C T2 G_def NE_def)
132
133
134 (* Consistency is confirmed by Nitpick *)
135 lemma True nitpick [satisfy, user_axioms] oops
136
```

Below the code, the output pane displays Skolem constants and constants defined in the theory:

```
Skolem constants:
λxa. ??G.x =
  (λx. _)(μ1 := (λx. _)((μ1, i1) := True, (μ1, i2) := False))
v = i2
w = i2

Constants:
K = (λx. _)
  ((i1, i1) := True, (i1, i2) := False, (i2, i1) := True,
   (i2, i2) := True)
P = (λx. _)
  (((λx. _)((μ1, i1) := True, (μ1, i2) := True), i1) := True,
   (((λx. _)((μ1, i1) := True, (μ1, i2) := True), i2) := True,
    (((λx. _)((μ1, i1) := True, (μ1, i2) := False), i1) := True,
     (((λx. _)((μ1, i1) := True, (μ1, i2) := False), i2) := True,
      (((λx. _)((μ1, i1) := False, (μ1, i2) := True), i1) := False,
```

The bottom status bar shows: 130,29 (4705/5009) Isabelle parsing complete... (isabelle,isabelle,UTF-8-Isabelle) N m r o UG 138/639MB 11:40 AM

Conclusion

HOL as a (quite) Universal Metalogic via **Shallow Semantic Embeddings**:

- ▶ Very **lean approach** to integrate and combine logics
- ▶ High degree of **automation** (via the embeddings)
- ▶ **Uniform proofs** (via the embedding embeddings)
- ▶ Works surprisingly well in (our recent) **practical applications**
- ▶ **Intuitive interaction** at abstract level supported by proof assistants
- ▶ **Novel results** in different application domains
- ▶ Approach well suited also for **teaching logics**

Approach has recently been used in my interdisciplinary **lecture course on Computational Metaphysics** (which won the central teaching award of FU Berlin)

- ▶ Quickly adopted by novice students
- ▶ Very good feedback from students
- ▶ Some impressive results from student projects