On the Logical Necessity of God and the Impossibility of the Devil in Gödel's Modal Ontological Argument

Insights from Experiments in Computational Metaphysics

Christoph Benzmüller and Luca Pasetto

U Bamberg | FU Berlin | U Luxemburg

ESSLLI 2026

(English version of a talk given by Benzmüller at the
Berlin-Brandenburg Academy of Sciences and Humanities, Berlin, July 30, 2026)

Introductory Remarks

Computational Metaphysics

Formal methods for the analysis of philosophical arguments

Analysis of philosophical arguments using formal methods

Logic-Pluralistic Knowledge Representation and Reasoning

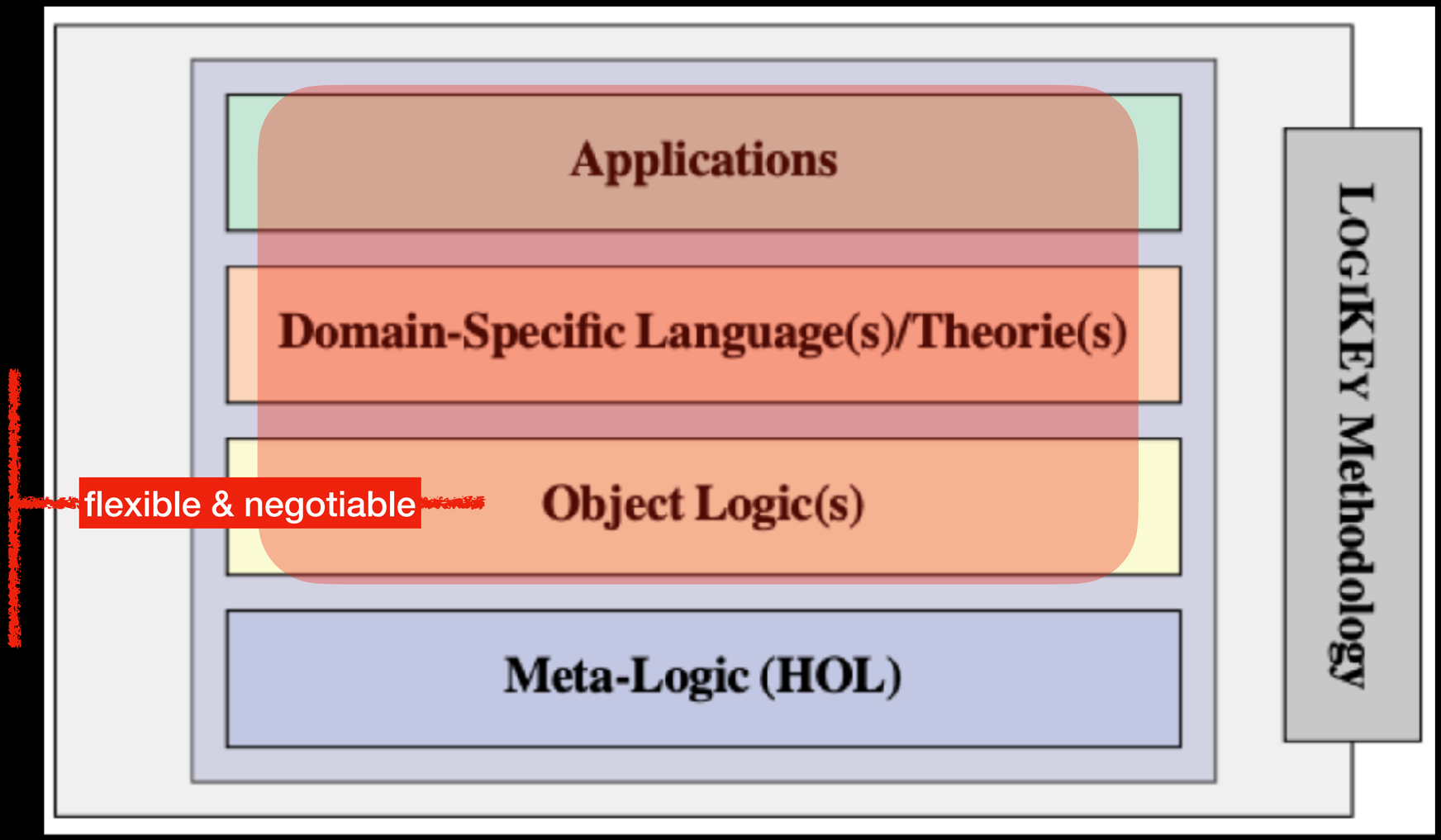
LogiKEy

Different modal logics:
K, T, KB, S4, S5, etc

Different quantifiers

Extensional or intensional

etc.



Origins: 2007-2008 (with L. Paulson)

Term LogiKEy: 2017-2019 (with L. van der Torre & X. Parent)

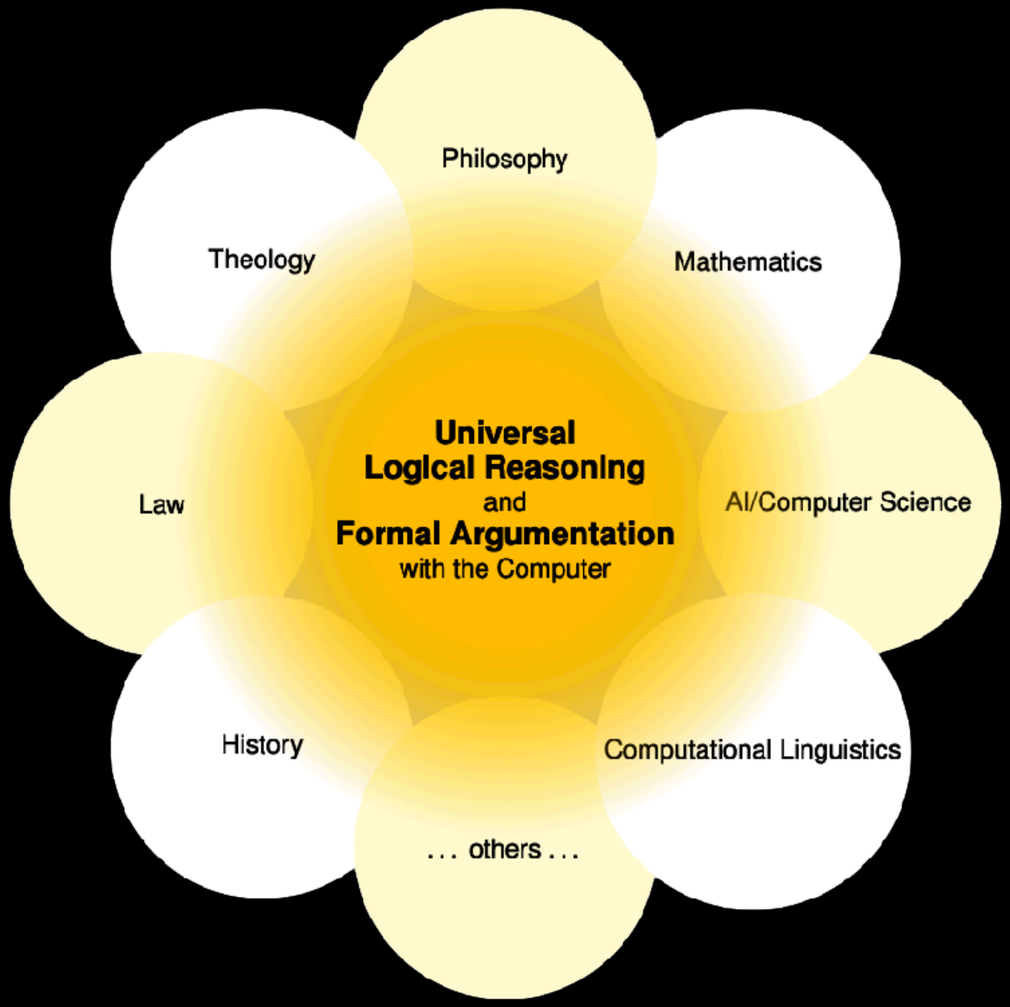
‘Exact’ methods are possible …

Es gibt syst<ematische> Methoden zur Lösung aller Probl<eme> (auch Kunst etc.).

Es gibt eine wissenschaftliche (exakte) Ph<ilosophie> {und Th<eologie>}, welche die Begriffe der höchsten Abstraktheit behandeln.

Kurt Gödel: Maximen V

Kurt Gödel (1906-78)



… far beyond the realm of mathematics!

Applying LogiKEy to Gödel's modal ontological argument

Work that I will not address here:

- Analysis of Scott's variant
- Analysis of Fitting's variant (extensions)
- Analysis of Anderson's variant
- Studies of related arguments
- Studies of (controversially) simplified variants

Types, Tableaus and Gödel's God in Isabelle/HOL 2017
David Fuenmayor, Christoph Benzmüller · *Archive of Formal Proofs (data publication)*, 1–34

Automating Emendations of the Ontological Argument in Intensional Higher-Order Modal Logic 2017
David Fuenmayor, Christoph Benzmüller · *KI 2017: Advances in Artificial Intelligence, 40th Annual German Conference on AI, Dortmund, Germany, September 25-29, 2017, Proceedings* pp. 114-127

Computer-supported Analysis of Positive Properties, Ultrafilters and Modal Collapse in Variants of Gödel's Ontological Argument 2020
Christoph Benzmüller, David Fuenmayor · *Bulletin of the Section of Logic* 49(2), 127-148

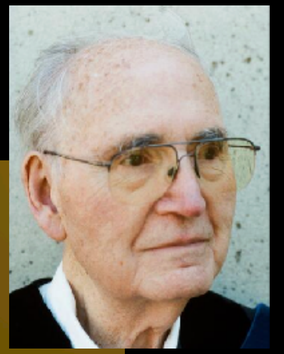
A Case Study On Computational Hermeneutics: E. J. Lowe's Modal Ontological Argument 2020
David Fuenmayor, Christoph Benzmüller · *Beyond Faith and Rationality: Essays on Logic, Religion and Philosophy*

A Simplified Variant of Gödel's Ontological Argument 2023
Christoph Benzmüller · *Beyond Babel: Religions and Linguistic Pluralism*

...

Important: In this talk we consider modal logic(s) as extension(s) of classical logic, type theory (Church), full comprehension, intensions of properties

Rest of this presentation is based on …



jww: Dana Scott

Automating Gödel's Ontological Proof of God's Existence with Higher-order Automated Theorem Provers 2014

Christoph Benzmüller, Bruno Woltzenlogel Paleo · *ECAI 2014* pp. 93 – 98

extended in

The Inconsistency in Gödel's Ontological Argument: A Success Story for AI in Metaphysics 2016

Christoph Benzmüller, Bruno Woltzenlogel Paleo · *IJCAI 2016* pp. 936-942

Home › Monatshefte für Mathematik › Article

Notes on Gödel's and Scott's variants of the ontological argument

Open access | Published: 21 April 2025

Volume 208, pages 569–611 (2025) Cite this article

You have full access to this open access article

Download PDF

Save article View saved research ›

Monatshefte für Mathematik

Aims and scope →

Submit manuscript →

Christoph Benzmüller & Dana Scott

7909 Accesses 4 Citations 4 Altmetric Explore all metrics →

Abstract

Notes on Kurt Gödel's modal ontological argument and Dana Scott's variant of it are presented. These remarks, supported by experimental studies with a proof assistant system for classical higher-order logic, implicitly answer some questions the authors have received over the last decade(s). In addition, some new insights resulting from the conducted experiments are reported.

extended in

arXiv Search Submit Donate Log in

Computer Science > Logic in Computer Science

[Submitted on 26 May 2026]

Many Logics, One Methodology: A Plea for Logical Pluralism in Formalised Reasoning (preprint)

Christoph Benzmüller, Daniel Kirchner, Luca Pasetto

This position statement looks back on two decades of work on shallow embeddings of non-classical logics in classical higher-order logic (HOL), a line of research that expanded into a range of logic embeddings in HOL and inspired the LogiKEy logic-pluralistic knowledge representation and reasoning methodology. This paper advances the case for logical pluralism at object-logic level within a unifying meta-logical framework such as LogiKEy, grounding the argument in computational metaphysics. More broadly, it advocates principled support for logical pluralism in modern proof assistants, and cautions against logical imperialism -- the rigid adoption of a single foundational logic for large-scale theory developments -- which impedes the interdisciplinary reuse that LogiKEy is designed to enable.

Access Paper:
View PDF
HTML (experimental)
TeX Source
view license

Current browse context:
cs.LO
< prev | next >
new | recent | 2026-05
Change to browse by:
cs
cs.AI
math
math.LO

References & Citations
NASA ADS
Google Scholar
Semantic Scholar

Export BibTeX Citation

Bookmark

… and some further, novel insights

Ontologischer Beweis

Feb 10, 1970

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

A1 $P(\varphi) \cdot P(\psi) \supset P(\varphi \cdot \psi)$ A2 $P(\varphi) \vee P(\sim\varphi)$

D1 $G(x) \equiv (\varphi)[P(\varphi) \supset \varphi(x)]$ (God)

D2 $\varphi \, Ess.\, x \equiv (\psi)[\psi(x) \supset_N (y)[\varphi(y) \supset \psi(y)]]$ (Essence of x)

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

A2 $P(\varphi) \supset N P(\varphi)$
$\sim P(\varphi) \supset N \sim P(\varphi)$
} because it follows from the nature of the property

Th. $G(x) \supset G\ Ess.\ x$

Df. $E(x) \equiv (\varphi)[\varphi\ Ess\ x \supset_N \exists x\, \varphi(x)]$ necessary Existen

Ax 3 $P(E)$

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

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

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

any two essences of x are nec. equivalent

exclusive or and for any number of summands

higher-order

modal

important!

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

A4: $P(\varphi) \cdot \varphi \supset_N \psi : \supset P(\psi)$ which imp

~~dence~~ $x = x$ is positive
~~ent~~ $x \neq x$ is negative

But if a system S of pos. propo. were in con
it would mean that the same prop. a (which
is positive) would be $x \neq x$

Positive means positive in the moral aesth
sense (independly of the accidental structure of
the world). Only then the ax. true. It m
also meant "attribution" as opposed to "privation
(or containg privation) - This interpret. implies prof

If φ positive not: $(x)\, N \sim \varphi(x)$. Otherwise $\varphi(x) \supset_N x \neq x$
hence $x \neq x$ positive not $x = x$ dig contrary A4
or the exist. of pos. attr.

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

Overview

- HOML in HOL
- ~~Scott's variant (with PG, as we already know it)~~
- Gödel's scriptum variant (1970)
  a. **Gödel1:** scriptum notion of essence — inconsistency
  b. **Gödel2:** modified notion of essence — $\Box\exists G$
  c. **Gödel3:** modified notion of property inclusion — $\Box\exists G$
  d. **No (perfect) Devil (or Devil = God)** — $\Box\exists G$
- Gödel's scriptum variant (1970) and mathematical realism is (uncountably/maximally) infinite
- Personal conclusion

new formal studies

new variant

— C

LogiKEy@work

HOML in HOL

HOML in HOL

Unimportant Isabelle/HOL parameter settings

```
theory HOMLinHOL imports Main
begin
—‹Global parameters setting for the model finder nitpick and the parser; unimport for the reader›
 nitpick_params[user_axioms,expect=genuine,show_all,format=2,max_genuine=3]
 declare[[syntax_ambiguity_warning=false]]
```

Fig. 3 Shallow embedding of higher-order modal logic (HOML) in the classical higher-order logic (HOL) of Isabelle/HOL utilizing the LogiKEy methodology

HOML in HOL

Type **i** for possible worlds
Type e for individuals
Type σ := (i=>bool) for modal formulas
Type τ := (e=>σ) for modal predicates
Constant r for the accessibility between worlds
r depending on the logic refl./trans./symm. (K, T, KB, S4, S5)

```
theory HOMLinHOL imports Main
begin
—‹Global parameters setting for the model finder nitpick and the parser; unimport for the reader›
 nitpick_params[user_axioms,expect=genuine,show_all,format=2,max_genuine=3]
 declare[[syntax_ambiguity_warning=false]]
—‹Type i is associated with possible worlds and type e with entities:›
 typedecl i    —‹Possible worlds›
 typedecl e    —‹Individuals/entities›
 type_synonym σ = "i⇒bool"    —‹World-lifted propositions›
 type_synonym τ = "e⇒σ"       —‹Modal properties›
 consts R::"i⇒i⇒bool" ("_r_")   —‹Accessibility relation between worlds›
 axiomatization where
   Rrefl: "∀x. xrx"   and   Rsymm: "∀x y. xry ⟶ yrx"   and   Rtrans: "∀x y z. xry ∧ yrz ⟶ xrz"
```

Fig. 3 Shallow embedding of higher-order modal logic (HOML) in the classical higher-order logic (HOL) of Isabelle/HOL utilizing the LogiKEy methodology

HOML in HOL

Modal logic connectives ⊥ ⊤ ¬ ∧ ∨ ⊃ ↔

```
theory HOMLinHOL imports Main
begin
—‹Global parameters setting for the model finder nitpick and the parser; unimport for the reader›
 nitpick_params[user_axioms,expect=genuine,show_all,format=2,max_genuine=3]
 declare[[syntax_ambiguity_warning=false]]
—‹Type i is associated with possible worlds and type e with entities:›
 typedecl i    —‹Possible worlds›
 typedecl e    —‹Individuals/entities›
 type_synonym σ = "i⇒bool"    —‹World-lifted propositions›
 type_synonym τ = "e⇒σ"       —‹Modal properties›
 consts R::"i⇒i⇒bool" ("_r_")   —‹Accessibility relation between worlds›
 axiomatization where
   Rrefl: "∀x. xrx"   and   Rsymm: "∀x y. xry ⟶ yrx"   and   Rtrans: "∀x y z. xry ∧ yrz ⟶ xrz"
—‹Logical connectives (operating on truth-sets):›
 abbreviation Mbot::σ ("⊥") where                "⊥ ≡ λw. False"
 abbreviation Mtop::σ ("⊤") where                "⊤ ≡ λw. True"
 abbreviation Mneg::"σ⇒σ" ("¬_" [52]53) where    "¬φ ≡ λw. ¬(φ w)"
 abbreviation Mand::"σ⇒σ⇒σ" (infixl "∧" 50) where   "φ∧ψ ≡ λw. φ w ∧ ψ w"
 abbreviation Mor::"σ⇒σ⇒σ" (infixl "∨" 49) where    "φ∨ψ ≡ λw. φ w ∨ ψ w "
 abbreviation Mimp::"σ⇒σ⇒σ" (infixr "⊃" 48) where   "φ⊃ψ ≡ λw. φ w ⟶ ψ w"
 abbreviation Mequiv::"σ⇒σ⇒σ" (infixl "↔" 47) where "φ↔ψ ≡ λw. φ w ⟷ ψ w"
```

Fig. 3 Shallow embedding of higher-order modal logic (HOML) in the classical higher-order logic (HOL) of Isabelle/HOL utilizing the LogiKEy methodology

HOML in HOL

Box and Diamond: □ ◊ (via accessibility r)

```
theory HOMLinHOL imports Main
begin
—‹Global parameters setting for the model finder nitpick and the parser; unimport for the reader›
 nitpick_params[user_axioms,expect=genuine,show_all,format=2,max_genuine=3]
 declare[[syntax_ambiguity_warning=false]]
—‹Type i is associated with possible worlds and type e with entities:›
 typedecl i    —‹Possible worlds›
 typedecl e    —‹Individuals/entities›
 type_synonym σ = "i⇒bool"    —‹World-lifted propositions›
 type_synonym τ = "e⇒σ"       —‹Modal properties›
 consts R::"i⇒i⇒bool" ("_r_")   —‹Accessibility relation between worlds›
 axiomatization where
   Rrefl: "∀x. xrx"   and   Rsymm: "∀x y. xry ⟶ yrx"   and   Rtrans: "∀x y z. xry ∧ yrz ⟶ xrz"
—‹Logical connectives (operating on truth-sets):›
 abbreviation Mbot::σ ("⊥") where          "⊥ ≡ λw. False"
 abbreviation Mtop::σ ("⊤") where          "⊤ ≡ λw. True"
 abbreviation Mneg::"σ⇒σ" ("¬_" [52]53) where   "¬φ ≡ λw. ¬(φ w)"
 abbreviation Mand::"σ⇒σ⇒σ" (infixl "∧" 50) where   "φ∧ψ ≡ λw. φ w ∧ ψ w"
 abbreviation Mor::"σ⇒σ⇒σ" (infixl "∨" 49) where   "φ∨ψ ≡ λw. φ w ∨ ψ w "
 abbreviation Mimp::"σ⇒σ⇒σ" (infixr "⊃" 48) where   "φ⊃ψ ≡ λw. φ w ⟶ ψ w"
 abbreviation Mequiv::"σ⇒σ⇒σ" (infixl "↔" 47) where   "φ↔ψ ≡ λw. φ w ⟷ ψ w"
 abbreviation Mbox::"σ⇒σ" ("□_" [54]55) where   "□φ ≡ λw.∀v. w r v ⟶ φ v"
 abbreviation Mdia::"σ⇒σ" ("◇_" [54]55) where   "◇φ ≡ λw.∃v. w r v ∧ φ v"
```

Fig. 3 Shallow embedding of higher-order modal logic (HOML) in the classical higher-order logic (HOL) of Isabelle/HOL utilizing the LogiKEy methodology

HOML in HOL

Primitive equality (rigid, polymorphic)

```
theory HOMLinHOL imports Main
begin
—‹Global parameters setting for the model finder nitpick and the parser; unimport for the reader›
 nitpick_params[user_axioms,expect=genuine,show_all,format=2,max_genuine=3]
 declare[[syntax_ambiguity_warning=false]]
—‹Type i is associated with possible worlds and type e with entities:›
 typedecl i    —‹Possible worlds›
 typedecl e    —‹Individuals/entities›
 type_synonym σ = "i⇒bool"    —‹World-lifted propositions›
 type_synonym τ = "e⇒σ"       —‹Modal properties›
 consts R::"i⇒i⇒bool" ("_r_")   —‹Accessibility relation between worlds›
 axiomatization where
   Rrefl: "∀x. xrx"   and   Rsymm: "∀x y. xry ⟶ yrx"   and   Rtrans: "∀x y z. xry ∧ yrz ⟶ xrz"
—‹Logical connectives (operating on truth-sets):›
 abbreviation Mbot::σ ("⊥") where                "⊥ ≡ λw. False"
 abbreviation Mtop::σ ("⊤") where                "⊤ ≡ λw. True"
 abbreviation Mneg::"σ⇒σ" ("¬_" [52]53) where    "¬φ ≡ λw. ¬(φ w)"
 abbreviation Mand::"σ⇒σ⇒σ" (infixl "∧" 50) where   "φ∧ψ ≡ λw. φ w ∧ ψ w"
 abbreviation Mor::"σ⇒σ⇒σ" (infixl "∨" 49) where    "φ∨ψ ≡ λw. φ w ∨ ψ w "
 abbreviation Mimp::"σ⇒σ⇒σ" (infixr "⊃" 48) where   "φ⊃ψ ≡ λw. φ w ⟶ ψ w"
 abbreviation Mequiv::"σ⇒σ⇒σ" (infixl "↔" 47) where "φ↔ψ ≡ λw. φ w ⟷ ψ w"
 abbreviation Mbox::"σ⇒σ" ("□_" [54]55) where    "□φ ≡ λw.∀v. w r v ⟶ φ v"
 abbreviation Mdia::"σ⇒σ" ("◇_" [54]55) where    "◇φ ≡ λw.∃v. w r v ∧ φ v"
 abbreviation Mprimeq::"'a⇒'a⇒σ" ("_=_") where   "x=y ≡ λw. x=y"
 abbreviation Mprimneg::"'a⇒'a⇒σ" ("_≠_") where  "x≠y ≡ λw. x≠y"
```

Fig. 3 Shallow embedding of higher-order modal logic (HOML) in the classical higher-order logic (HOL) of Isabelle/HOL utilizing the LogiKEy methodology

HOML in HOL

Complement & conjunction of modal properties, and exclusive-or

```
theory HOMLinHOL imports Main
begin
—‹Global parameters setting for the model finder nitpick and the parser; unimport for the reader›
 nitpick_params[user_axioms,expect=genuine,show_all,format=2,max_genuine=3]
 declare[[syntax_ambiguity_warning=false]]
—‹Type i is associated with possible worlds and type e with entities:›
 typedecl i    —‹Possible worlds›
 typedecl e    —‹Individuals/entities›
 type_synonym σ = "i⇒bool"    —‹World-lifted propositions›
 type_synonym τ = "e⇒σ"       —‹Modal properties›
 consts R::"i⇒i⇒bool" ("_r_")  —‹Accessibility relation between worlds›
 axiomatization where
   Rrefl: "∀x. xrx"   and   Rsymm: "∀x y. xry ⟶ yrx"   and   Rtrans: "∀x y z. xry ∧ yrz ⟶ xrz"
—‹Logical connectives (operating on truth-sets):›
 abbreviation Mbot::σ ("⊥") where                  "⊥ ≡ λw. False"
 abbreviation Mtop::σ ("⊤") where                  "⊤ ≡ λw. True"
 abbreviation Mneg::"σ⇒σ" ("¬_" [52]53) where      "¬φ ≡ λw. ¬(φ w)"
 abbreviation Mand::"σ⇒σ⇒σ" (infixl "∧" 50) where  "φ∧ψ ≡ λw. φ w ∧ ψ w"
 abbreviation Mor::"σ⇒σ⇒σ" (infixl "∨" 49) where   "φ∨ψ ≡ λw. φ w ∨ ψ w "
 abbreviation Mimp::"σ⇒σ⇒σ" (infixr "⊃" 48) where  "φ⊃ψ ≡ λw. φ w ⟶ ψ w"
 abbreviation Mequiv::"σ⇒σ⇒σ" (infixl "↔" 47) where "φ↔ψ ≡ λw. φ w ⟷ ψ w"
 abbreviation Mbox::"σ⇒σ" ("□_" [54]55) where     "□φ ≡ λw.∀v. w r v ⟶ φ v"
 abbreviation Mdia::"σ⇒σ" ("◇_" [54]55) where     "◇φ ≡ λw.∃v. w r v ∧ φ v"
 abbreviation Mprimeq::"'a⇒'a⇒σ" ("_=_") where    "x=y ≡ λw. x=y"
 abbreviation Mprimneq::"'a⇒'a⇒σ" ("_≠_") where   "x≠y ≡ λw. x≠y"
 abbreviation Mnegpred::"τ⇒τ" ("~_") where        "~Φ ≡ λx.λw. ¬Φ x w"
 abbreviation Mconpred::"τ⇒τ⇒τ" (infixl "." 50) where "Φ.Ψ ≡ λx.λw. Φ x w ∧ Ψ x w"
 abbreviation Mexclor::"σ⇒σ⇒σ" (infixl "∨ᵉ" 49) where "φ∨ᵉψ ≡ (φ ∨ ψ) ∧ ¬(φ ∧ ψ)"
```

Fig. 3 Shallow embedding of higher-order modal logic (HOML) in the classical higher-order logic (HOL) of Isabelle/HOL utilizing the LogiKEy methodology

HOML in HOL

Quantifiers (constant domains, polymorphic)

```
theory HOMLinHOL imports Main
begin
—‹Global parameters setting for the model finder nitpick and the parser; unimport for the reader›
 nitpick_params[user_axioms,expect=genuine,show_all,format=2,max_genuine=3]
 declare[[syntax_ambiguity_warning=false]]
—‹Type i is associated with possible worlds and type e with entities:›
 typedecl i    —‹Possible worlds›
 typedecl e    —‹Individuals/entities›
 type_synonym σ = "i⇒bool"    —‹World-lifted propositions›
 type_synonym τ = "e⇒σ"       —‹Modal properties›
 consts R::"i⇒i⇒bool" ("_r_")   —‹Accessibility relation between worlds›
 axiomatization where
   Rrefl: "∀x. xrx"   and   Rsymm: "∀x y. xry ⟶ yrx"   and   Rtrans: "∀x y z. xry ∧ yrz ⟶ xrz"
—‹Logical connectives (operating on truth-sets):›
 abbreviation Mbot::σ ("⊥") where                "⊥ ≡ λw. False"
 abbreviation Mtop::σ ("⊤") where                "⊤ ≡ λw. True"
 abbreviation Mneg::"σ⇒σ" ("¬_" [52]53) where    "¬φ ≡ λw. ¬(φ w)"
 abbreviation Mand::"σ⇒σ⇒σ" (infixl "∧" 50) where   "φ∧ψ ≡ λw. φ w ∧ ψ w"
 abbreviation Mor::"σ⇒σ⇒σ" (infixl "∨" 49) where    "φ∨ψ ≡ λw. φ w ∨ ψ w "
 abbreviation Mimp::"σ⇒σ⇒σ" (infixr "⊃" 48) where   "φ⊃ψ ≡ λw. φ w ⟶ ψ w"
 abbreviation Mequiv::"σ⇒σ⇒σ" (infixl "↔" 47) where "φ↔ψ ≡ λw. φ w ⟷ ψ w"
 abbreviation Mbox::"σ⇒σ" ("□_" [54]55) where    "□φ ≡ λw.∀v. w r v ⟶ φ v"
 abbreviation Mdia::"σ⇒σ" ("◇_" [54]55) where    "◇φ ≡ λw.∃v. w r v ∧ φ v"
 abbreviation Mprimeq::"'a⇒'a⇒σ" ("_=_") where   "x=y ≡ λw. x=y"
 abbreviation Mprimneq::"'a⇒'a⇒σ" ("_≠_") where  "x≠y ≡ λw. x≠y"
 abbreviation Mnegpred::"τ⇒τ" ("~_") where       "~Φ ≡ λx.λw. ¬Φ x w"
 abbreviation Mconpred::"τ⇒τ⇒τ" (infixl "." 50) where  "Φ.Ψ ≡ λx.λw. Φ x w ∧ Ψ x w"
 abbreviation Mexclor::"σ⇒σ⇒σ" (infixl "∨ᵉ" 49) where  "φ∨ᵉψ ≡ (φ ∨ ψ) ∧ ¬(φ ∧ ψ)"
—‹Possibilist quantifiers (polymorphic)›
 abbreviation Mallposs::"('a⇒σ)⇒σ" ("∀") where   "∀Φ ≡ λw.∀x. Φ x w"
 abbreviation Mallpossb (binder "∀" [8]9) where  "∀x. φ(x) ≡ ∀φ"
 abbreviation Mexiposs::"('a⇒σ)⇒σ" ("∃") where   "∃Φ ≡ λw.∃x. Φ x w"
 abbreviation Mexipossb (binder "∃" [8]9) where  "∃x. φ(x) ≡ ∃φ"
```

Fig. 3 Shallow embedding of higher-order modal logic (HOML) in the classical higher-order logic (HOL) of Isabelle/HOL utilizing the LogiKEy methodology

HOML in HOL

Quantifiers (varying domains, for individuals)
(x@w models the existence of x in world w)

```
theory HOMLinHOL imports Main
begin
—‹Global parameters setting for the model finder nitpick and the parser; unimport for the reader›
 nitpick_params[user_axioms,expect=genuine,show_all,format=2,max_genuine=3]
 declare[[syntax_ambiguity_warning=false]]
—‹Type i is associated with possible worlds and type e with entities:›
 typedecl i    —‹Possible worlds›
 typedecl e    —‹Individuals/entities›
 type_synonym σ = "i⇒bool"    —‹World-lifted propositions›
 type_synonym τ = "e⇒σ"    —‹Modal properties›
 consts R::"i⇒i⇒bool" ("_r_")    —‹Accessibility relation between worlds›
 axiomatization where
   Rrefl: "∀x. xrx"   and   Rsymm: "∀x y. xry ⟶ yrx"   and   Rtrans: "∀x y z. xry ∧ yrz ⟶ xrz"
—‹Logical connectives (operating on truth-sets):›
 abbreviation Mbot::σ ("⊥") where                    "⊥ ≡ λw. False"
 abbreviation Mtop::σ ("⊤") where                    "⊤ ≡ λw. True"
 abbreviation Mneg::"σ⇒σ" ("¬_" [52]53) where        "¬φ ≡ λw. ¬(φ w)"
 abbreviation Mand::"σ⇒σ⇒σ" (infixl "∧" 50) where    "φ∧ψ ≡ λw. φ w ∧ ψ w"
 abbreviation Mor::"σ⇒σ⇒σ" (infixl "∨" 49) where     "φ∨ψ ≡ λw. φ w ∨ ψ w "
 abbreviation Mimp::"σ⇒σ⇒σ" (infixr "⊃" 48) where    "φ⊃ψ ≡ λw. φ w ⟶ ψ w"
 abbreviation Mequiv::"σ⇒σ⇒σ" (infixl "↔" 47) where  "φ↔ψ ≡ λw. φ w ⟷ ψ w"
 abbreviation Mbox::"σ⇒σ" ("□_" [54]55) where        "□φ ≡ λw.∀v. w r v ⟶ φ v"
 abbreviation Mdia::"σ⇒σ" ("◇_" [54]55) where        "◇φ ≡ λw.∃v. w r v ∧ φ v"
 abbreviation Mprimeq::"'a⇒'a⇒σ" ("_=_") where       "x=y ≡ λw. x=y"
 abbreviation Mprimneg::"'a⇒'a⇒σ" ("_≠_") where      "x≠y ≡ λw. x≠y"
 abbreviation Mnegpred::"τ⇒τ" ("~_") where           "~Φ ≡ λx.λw. ¬Φ x w"
 abbreviation Mconpred::"τ⇒τ⇒τ" (infixl "." 50) where "Φ.Ψ ≡ λx.λw. Φ x w ∧ Ψ x w"
 abbreviation Mexclor::"σ⇒σ⇒σ" (infixl "∨ᵉ" 49) where "φ∨ᵉψ ≡ (φ ∨ ψ) ∧ ¬(φ ∧ ψ)"
—‹Possibilist quantifiers (polymorphic)›
 abbreviation Mallposs::"('a⇒σ)⇒σ" ("∀") where      "∀Φ ≡ λw.∀x. Φ x w"
 abbreviation Mallpossb (binder "∀" [8]9) where     "∀x. φ(x) ≡ ∀φ"
 abbreviation Mexiposs::"('a⇒σ)⇒σ" ("∃") where      "∃Φ ≡ λw.∃x. Φ x w"
 abbreviation Mexipossb (binder "∃" [8]9) where     "∃x. φ(x) ≡ ∃φ"
—‹Actualist quantifiers (for individuals/entities)›
 consts existsAt::"e⇒σ" ("_@_")
 abbreviation Mallact::"(e⇒σ)⇒σ" ("∀ᴱ") where       "∀ᴱΦ ≡ λw.∀x. x@w ⟶ Φ x w"
 abbreviation Mallactb (binder "∀ᴱ" [8]9) where     "∀ᴱx. φ(x) ≡ ∀ᴱφ"
 abbreviation Mexiact::"(e⇒σ)⇒σ" ("∃ᴱ") where      "∃ᴱΦ ≡ λw.∃x. x@w ∧ Φ x w"
 abbreviation Mexiactb (binder "∃ᴱ" [8]9) where     "∃ᴱx. φ(x) ≡ ∃ᴱφ"
```

Fig. 3 Shallow embedding of higher-order modal logic (HOML) in the classical higher-order logic (HOL) of Isabelle/HOL utilizing the LogiKEy methodology

HOML in HOL


```
theory HOMLinHOL imports Main
begin
—‹Global parameters setting for the model finder nitpick and the parser; unimport for the reader›
 nitpick_params[user_axioms,expect=genuine,show_all,format=2,max_genuine=3]
 declare[[syntax_ambiguity_warning=false]]
—‹Type i is associated with possible worlds and type e with entities:›
 typedecl i    —‹Possible worlds›
 typedecl e    —‹Individuals/entities›
 type_synonym σ = "i⇒bool"    —‹World-lifted propositions›
 type_synonym τ = "e⇒σ"       —‹Modal properties›
 consts R::"i⇒i⇒bool" ("_r_")   —‹Accessibility relation between worlds›
 axiomatization where
   Rrefl: "∀x. xrx"   and   Rsymm: "∀x y. xry ⟶ yrx"   and   Rtrans: "∀x y z. xry ∧ yrz ⟶ xrz"
—‹Logical connectives (operating on truth-sets):›
 abbreviation Mbot::σ ("⊥") where                "⊥ ≡ λw. False"
 abbreviation Mtop::σ ("⊤") where                "⊤ ≡ λw. True"
 abbreviation Mneg::"σ⇒σ" ("¬_" [52]53) where    "¬φ ≡ λw. ¬(φ w)"
 abbreviation Mand::"σ⇒σ⇒σ" (infixl "∧" 50) where   "φ∧ψ ≡ λw. φ w ∧ ψ w"
 abbreviation Mor::"σ⇒σ⇒σ" (infixl "∨" 49) where    "φ∨ψ ≡ λw. φ w ∨ ψ w "
 abbreviation Mimp::"σ⇒σ⇒σ" (infixr "⊃" 48) where   "φ⊃ψ ≡ λw. φ w ⟶ ψ w"
 abbreviation Mequiv::"σ⇒σ⇒σ" (infixl "↔" 47) where "φ↔ψ ≡ λw. φ w ⟷ ψ w"
 abbreviation Mbox::"σ⇒σ" ("□_" [54]55) where    "□φ ≡ λw.∀v. w r v ⟶ φ v"
 abbreviation Mdia::"σ⇒σ" ("◇_" [54]55) where    "◇φ ≡ λw.∃v. w r v ∧ φ v"
 abbreviation Mprimeq::"'a⇒'a⇒σ" ("_=_") where   "x=y ≡ λw. x=y"
 abbreviation Mprimneq::"'a⇒'a⇒σ" ("_≠_") where  "x≠y ≡ λw. x≠y"
 abbreviation Mnegpred::"τ⇒τ" ("~_") where       "~Φ ≡ λx.λw. ¬Φ x w"
 abbreviation Mconpred::"τ⇒τ⇒τ" (infixl "." 50) where  "Φ.Ψ ≡ λx.λw. Φ x w ∧ Ψ x w"
 abbreviation Mexclor::"σ⇒σ⇒σ" (infixl "∨ᵉ" 49) where  "φ∨ᵉψ ≡ (φ ∨ ψ) ∧ ¬(φ ∧ ψ)"
—‹Possibilist quantifiers (polymorphic)›
 abbreviation Mallposs::"('a⇒σ)⇒σ" ("∀") where     "∀Φ ≡ λw.∀x. Φ x w"
 abbreviation Mallpossb (binder "∀" [8]9) where   "∀x. φ(x) ≡ ∀φ"
 abbreviation Mexiposs::"('a⇒σ)⇒σ" ("∃") where     "∃Φ ≡ λw.∃x. Φ x w"
 abbreviation Mexipossb (binder "∃" [8]9) where   "∃x. φ(x) ≡ ∃φ"
—‹Actualist quantifiers (for individuals/entities)›
 consts existsAt::"e⇒σ" ("_@_")
 abbreviation Mallact::"(e⇒σ)⇒σ" ("∀ᴱ") where     "∀ᴱΦ ≡ λw.∀x. x@w ⟶ Φ x w"
 abbreviation Mallactb (binder "∀ᴱ" [8]9) where   "∀ᴱx. φ(x) ≡ ∀ᴱφ"
 abbreviation Mexiact::"(e⇒σ)⇒σ" ("∃ᴱ") where     "∃ᴱΦ ≡ λw.∃x. x@w ∧ Φ x w"
 abbreviation Mexiactb (binder "∃ᴱ" [8]9) where   "∃ᴱx. φ(x) ≡ ∃ᴱφ"
—‹Leibniz equality (polymorphic):›
 abbreviation Mleibeq::"'a⇒'a⇒σ" ("_≡_") where    "x≡y ≡ ∀P. P x ⊃ P y"
```

Leibniz equality (polymorphic)

Fig. 3 Shallow embedding of higher-order modal logic (HOML) in the classical higher-order logic (HOL) of Isabelle/HOL utilizing the LogiKEy methodology

HOML in HOL

Notion of validity for modal formulas

```
theory HOMLinHOL imports Main
begin
—‹Global parameters setting for the model finder nitpick and the parser; unimport for the reader›
 nitpick_params[user_axioms,expect=genuine,show_all,format=2,max_genuine=3]
 declare[[syntax_ambiguity_warning=false]]
—‹Type i is associated with possible worlds and type e with entities:›
 typedecl i    —‹Possible worlds›
 typedecl e    —‹Individuals/entities›
 type_synonym σ = "i⇒bool"    —‹World-lifted propositions›
 type_synonym τ = "e⇒σ"    —‹Modal properties›
 consts R::"i⇒i⇒bool" ("_r_")    —‹Accessibility relation between worlds›
 axiomatization where
   Rrefl: "∀x. xrx"   and   Rsymm: "∀x y. xry ⟶ yrx"   and   Rtrans: "∀x y z. xry ∧ yrz ⟶ xrz"
—‹Logical connectives (operating on truth-sets):›
 abbreviation Mbot::σ ("⊥") where                "⊥ ≡ λw. False"
 abbreviation Mtop::σ ("⊤") where                "⊤ ≡ λw. True"
 abbreviation Mneg::"σ⇒σ" ("¬_" [52]53) where    "¬φ ≡ λw. ¬(φ w)"
 abbreviation Mand::"σ⇒σ⇒σ" (infixl "∧" 50) where   "φ∧ψ ≡ λw. φ w ∧ ψ w"
 abbreviation Mor::"σ⇒σ⇒σ" (infixl "∨" 49) where    "φ∨ψ ≡ λw. φ w ∨ ψ w "
 abbreviation Mimp::"σ⇒σ⇒σ" (infixr "⊃" 48) where   "φ⊃ψ ≡ λw. φ w ⟶ ψ w"
 abbreviation Mequiv::"σ⇒σ⇒σ" (infixl "↔" 47) where "φ↔ψ ≡ λw. φ w ⟷ ψ w"
 abbreviation Mbox::"σ⇒σ" ("□_" [54]55) where    "□φ ≡ λw.∀v. w r v ⟶ φ v"
 abbreviation Mdia::"σ⇒σ" ("◇_" [54]55) where    "◇φ ≡ λw.∃v. w r v ∧ φ v"
 abbreviation Mprimeq::"'a⇒'a⇒σ" ("_=_") where   "x=y ≡ λw. x=y"
 abbreviation Mprimneg::"'a⇒'a⇒σ" ("_≠_") where  "x≠y ≡ λw. x≠y"
 abbreviation Mnegpred::"τ⇒τ" ("~_") where       "~Φ ≡ λx.λw. ¬Φ x w"
 abbreviation Mconpred::"τ⇒τ⇒τ" (infixl "." 50) where  "Φ.Ψ ≡ λx.λw. Φ x w ∧ Ψ x w"
 abbreviation Mexclor::"σ⇒σ⇒σ" (infixl "∨ᵉ" 49) where  "φ∨ᵉψ ≡ (φ ∨ ψ) ∧ ¬(φ ∧ ψ)"
—‹Possibilist quantifiers (polymorphic)›
 abbreviation Mallposs::"('a⇒σ)⇒σ" ("∀") where   "∀Φ ≡ λw.∀x. Φ x w"
 abbreviation Mallpossb (binder "∀" [8]9) where  "∀x. φ(x) ≡ ∀φ"
 abbreviation Mexiposs::"('a⇒σ)⇒σ" ("∃") where   "∃Φ ≡ λw.∃x. Φ x w"
 abbreviation Mexipossb (binder "∃" [8]9) where  "∃x. φ(x) ≡ ∃φ"
—‹Actualist quantifiers (for individuals/entities)›
 consts existsAt::"e⇒σ" ("_@_")
 abbreviation Mallact::"(e⇒σ)⇒σ" ("∀ᴱ") where   "∀ᴱΦ ≡ λw.∀x. x@w ⟶ Φ x w"
 abbreviation Mallactb (binder "∀ᴱ" [8]9) where  "∀ᴱx. φ(x) ≡ ∀ᴱφ"
 abbreviation Mexiact::"(e⇒σ)⇒σ" ("∃ᴱ") where   "∃ᴱΦ ≡ λw.∃x. x@w ∧ Φ x w"
 abbreviation Mexiactb (binder "∃ᴱ" [8]9) where  "∃ᴱx. φ(x) ≡ ∃ᴱφ"
—‹Leibniz equality (polymorphic):›
 abbreviation Mleibeq::"'a⇒'a⇒σ" ("_≡_") where   "x≡y ≡ ∀P. P x ⊃ P y"
—‹Meta-logical predicate for global validity:›
 abbreviation Mvalid::"σ⇒bool" ("⌊_⌋") where     "⌊ψ⌋ ≡ ∀w. ψ w"
end
```

Fig. 3 Shallow embedding of higher-order modal logic (HOML) in the classical higher-order logic (HOL) of Isabelle/HOL utilizing the LogiKEy methodology

HOML in HOL

A mathematician would only look at these equations

```
theory HOMLinHOL imports Main
begin
—‹Global parameters setting for the model finder nitpick and the parser; unimport for the reader›
 nitpick_params[user_axioms,expect=genuine,show_all,format=2,max_genuine=3]
 declare[[syntax_ambiguity_warning=false]]
—‹Type i is associated with possible worlds and type e with entities:›
 typedecl i    —‹Possible worlds›
 typedecl e    —‹Individuals/entities›
 type_synonym σ = "i⇒bool"    —‹World-lifted propositions›
 type_synonym τ = "e⇒σ"    —‹Modal properties›
 consts R::"i⇒i⇒bool" ("_r_")    —‹Accessibility relation between worlds›
 axiomatization where
   Rrefl: "∀x. xrx"   and   Rsymm: "∀x y. xry ⟶ yrx"   and   Rtrans: "∀x y z. xry ∧ yrz ⟶ xrz"
—‹Logical connectives (operating on truth-sets):›
 abbreviation Mbot::σ ("⊥") where                "⊥ ≡ λw. False"
 abbreviation Mtop::σ ("⊤") where                "⊤ ≡ λw. True"
 abbreviation Mneg::"σ⇒σ" ("¬_" [52]53) where    "¬φ ≡ λw. ¬(φ w)"
 abbreviation Mand::"σ⇒σ⇒σ" (infixl "∧" 50) where    "φ∧ψ ≡ λw. φ w ∧ ψ w"
 abbreviation Mor::"σ⇒σ⇒σ" (infixl "∨" 49) where    "φ∨ψ ≡ λw. φ w ∨ ψ w "
 abbreviation Mimp::"σ⇒σ⇒σ" (infixr "⊃" 48) where    "φ⊃ψ ≡ λw. φ w ⟶ ψ w"
 abbreviation Mequiv::"σ⇒σ⇒σ" (infixl "↔" 47) where    "φ↔ψ ≡ λw. φ w ⟷ ψ w"
 abbreviation Mbox::"σ⇒σ" ("□_" [54]55) where    "□φ ≡ λw.∀v. w r v ⟶ φ v"
 abbreviation Mdia::"σ⇒σ" ("◇_" [54]55) where    "◇φ ≡ λw.∃v. w r v ∧ φ v"
 abbreviation Mprimeq::"'a⇒'a⇒σ" ("_=_") where    "x=y ≡ λw. x=y"
 abbreviation Mprimneq::"'a⇒'a⇒σ" ("_≠_") where    "x≠y ≡ λw. x≠y"
 abbreviation Mnegpred::"τ⇒τ" ("~_") where    "~Φ ≡ λx.λw. ¬Φ x w"
 abbreviation Mconpred::"τ⇒τ⇒τ" (infixl "." 50) where    "Φ.Ψ ≡ λx.λw. Φ x w ∧ Ψ x w"
 abbreviation Mexclor::"σ⇒σ⇒σ" (infixl "∨ᵉ" 49) where    "φ∨ᵉψ ≡ (φ ∨ ψ) ∧ ¬(φ ∧ ψ)"
—‹Possibilist quantifiers (polymorphic)›
 abbreviation Mallposs::"('a⇒σ)⇒σ" ("∀") where    "∀Φ ≡ λw.∀x. Φ x w"
 abbreviation Mallpossb (binder "∀" [8]9) where    "∀x. φ(x) ≡ ∀φ"
 abbreviation Mexiposs::"('a⇒σ)⇒σ" ("∃") where    "∃Φ ≡ λw.∃x. Φ x w"
 abbreviation Mexipossb (binder "∃" [8]9) where    "∃x. φ(x) ≡ ∃φ"
—‹Actualist quantifiers (for individuals/entities)›
 consts existsAt::"e⇒σ" ("_@_")
 abbreviation Mallact::"(e⇒σ)⇒σ" ("∀ᴱ") where    "∀ᴱΦ ≡ λw.∀x. x@w ⟶ Φ x w"
 abbreviation Mallactb (binder "∀ᴱ" [8]9) where    "∀ᴱx. φ(x) ≡ ∀ᴱφ"
 abbreviation Mexiact::"(e⇒σ)⇒σ" ("∃ᴱ") where    "∃ᴱΦ ≡ λw.∃x. x@w ∧ Φ x w"
 abbreviation Mexiactb (binder "∃ᴱ" [8]9) where    "∃ᴱx. φ(x) ≡ ∃ᴱφ"
—‹Leibniz equality (polymorphic):›
 abbreviation Mleibeq::"'a⇒'a⇒σ" ("_≡_") where    "x≡y ≡ ∀P. P x ⊃ P y"
—‹Meta-logical predicate for global validity:›
 abbreviation Mvalid::"σ⇒bool" ("⌊_⌋") where    "⌊ψ⌋ ≡ ∀w. ψ w"
end
```

Fig. 3 Shallow embedding of higher-order modal logic (HOML) in the classical higher-order logic (HOL) of Isabelle/HOL utilizing the LogiKEy methodology

HOML in HOL: tests

S5 axioms: valid as expected

```
theory TestsHOML imports HOMLinHOL
begin
—‹Test for S5 modal logic›
  lemma axM: "⌊□φ ⊃ φ⌋"          using Rrefl by blast
  lemma axD: "⌊□φ ⊃ ◇φ⌋"         using Rrefl by blast
  lemma axB: "⌊φ ⊃ □◇φ⌋"         using Rsymm by blast
  lemma ax4: "⌊□φ ⊃ □□φ⌋"        using Rtrans by blast
  lemma ax5: "⌊◇φ ⊃ □◇φ⌋"        using Rsymm Rtrans by blast
```

Fig. 4 Tests and verifications of properties for the embedding of HOML in HOL from Fig. 3

HOML in HOL: tests

S5 axioms: valid as expected
Barcan formulas: not valid for ’varying-dom.’ quant.
Barcan formulas: valid for ‘constant-dom.’ quant.

```
theory TestsHOML imports HOMLinHOL
begin
—‹Test for S5 modal logic›
  lemma axM: "⌊□φ ⊃ φ⌋"                    using Rrefl by blast
  lemma axD: "⌊□φ ⊃ ◇φ⌋"                   using Rrefl by blast
  lemma axB: "⌊φ ⊃ □◇φ⌋"                   using Rsymm by blast
  lemma ax4: "⌊□φ ⊃ □□φ⌋"                  using Rtrans by blast
  lemma ax5: "⌊◇φ ⊃ □◇φ⌋"          using Rsymm Rtrans by blast
—‹Test for Barcan and converse Barcan formula:›
  lemma BarcanAct:     "⌊(∀ᴱx.□(φ x)) ⊃ □(∀ᴱx.(φ x))⌋"   nitpick oops —‹Countermodel found›
  lemma ConvBarcanAct: "⌊□(∀ᴱx.(φ x)) ⊃ (∀ᴱx.□(φ x))⌋"   nitpick oops —‹Countermodel found›
  lemma BarcanPoss:    "⌊(∀x.□(φ x)) ⊃ □(∀x. φ x)⌋"     by blast
  lemma ConvBarcanPoss: "⌊□(∀x.(φ x)) ⊃ (∀x.□(φ x))⌋"    by blast
```

Fig. 4 Tests and verifications of properties for the embedding of HOML in HOL from Fig. 3

HOML in HOL: tests

S5 axioms: valid as expected
Barcan formulas: not valid for ’varying-dom.’ quant.
Barcan formulas: valid for ‘constant-dom.’ quant.

Hilbert axioms and rules: valid as expected

```
theory TestsHOML imports HOMLinHOL
begin
—‹Test for S5 modal logic›
  lemma axM: "⌊□φ ⊃ φ⌋"                    using Rrefl by blast
  lemma axD: "⌊□φ ⊃ ◇φ⌋"                   using Rrefl by blast
  lemma axB: "⌊φ ⊃ □◇φ⌋"                   using Rsymm by blast
  lemma ax4: "⌊□φ ⊃ □□φ⌋"                  using Rtrans by blast
  lemma ax5: "⌊◇φ ⊃ □◇φ⌋"          using Rsymm Rtrans by blast
—‹Test for Barcan and converse Barcan formula:›
  lemma BarcanAct:    "⌊(∀ᴱx.□(φ x)) ⊃ □(∀ᴱx.(φ x))⌋"   nitpick oops —‹Countermodel found›
  lemma ConvBarcanAct: "⌊□(∀ᴱx.(φ x)) ⊃ (∀ᴱx.□(φ x))⌋"  nitpick oops —‹Countermodel found›
  lemma BarcanPoss:   "⌊(∀x.□(φ x)) ⊃ □(∀x. φ x)⌋"     by blast
  lemma ConvBarcanPoss: "⌊□(∀x.(φ x)) ⊃ (∀x.□(φ x))⌋"  by blast
—‹A simple Hilbert system for classical propositional logic is derived›
  lemma Hilbert_A1: "⌊A ⊃ (B ⊃ A)⌋"                          by blast
  lemma Hilbert_A2: "⌊(A ⊃ (B ⊃ C)) ⊃ ((A ⊃ B) ⊃ (A ⊃ C))⌋"  by blast
  lemma Hilbert_MP: assumes "⌊A⌋" and "⌊A ⊃ B⌋" shows "⌊B⌋"  using assms by blast
```

Fig. 4 Tests and verifications of properties for the embedding of HOML in HOL from Fig. 3

HOML in HOL: tests

S5 axioms: valid as expected
Barcan formulas: not valid for ’varying-dom.’ quant.
Barcan formulas: valid for ‘constant-dom.’ quant.

Hilbert axioms and rules: valid as expected

Existential import: yes, for ‘constant-domain’ quant.

```
theory TestsHOML imports HOMLinHOL
begin
—‹Test for S5 modal logic›
  lemma axM: "⌊□φ ⊃ φ⌋"                    using Rrefl by blast
  lemma axD: "⌊□φ ⊃ ◇φ⌋"                   using Rrefl by blast
  lemma axB: "⌊φ ⊃ □◇φ⌋"                   using Rsymm by blast
  lemma ax4: "⌊□φ ⊃ □□φ⌋"                  using Rtrans by blast
  lemma ax5: "⌊◇φ ⊃ □◇φ⌋"          using Rsymm Rtrans by blast
—‹Test for Barcan and converse Barcan formula:›
  lemma BarcanAct:    "⌊(∀ᴱx.□(φ x)) ⊃ □(∀ᴱx.(φ x))⌋"   nitpick oops —‹Countermodel found›
  lemma ConvBarcanAct: "⌊□(∀ᴱx.(φ x)) ⊃ (∀ᴱx.□(φ x))⌋"  nitpick oops —‹Countermodel found›
  lemma BarcanPoss:   "⌊(∀x.□(φ x)) ⊃ □(∀x. φ x)⌋"      by blast
  lemma ConvBarcanPoss: "⌊□(∀x.(φ x)) ⊃ (∀x.□(φ x))⌋"   by blast
—‹A simple Hilbert system for classical propositional logic is derived›
  lemma Hilbert_A1: "⌊A ⊃ (B ⊃ A)⌋"                     by blast
  lemma Hilbert_A2: "⌊(A ⊃ (B ⊃ C)) ⊃ ((A ⊃ B) ⊃ (A ⊃ C))⌋"  by blast
  lemma Hilbert_MP: assumes "⌊A⌋" and "⌊A ⊃ B⌋" shows "⌊B⌋"  using assms by blast
—‹We have a polymorphic possibilist quantifier for which existential import holds›
  lemma Quant_1: assumes "⌊A⌋" shows "⌊∀x::'a. A⌋"   using assms by auto
—‹Existential import holds for possibilist quantifiers›
  lemma EximPossibilist1: "⌊∃x::e. x = x⌋"           by blast
  lemma EximPossibilist2: "⌊∃x::e. x ≡ x⌋"           by blast
  lemma EximPossibilist3: "⌊∃x::e. x = t⌋"           by blast
  lemma EximPossibilist4: "⌊∃x::'a. x ≡ t::'a⌋"      by blast
  lemma EximPossibilist:  "⌊∃x::'a. T⌋"              by blast
```

Fig. 4 Tests and verifications of properties for the embedding of HOML in HOL from Fig. 3

HOML in HOL: tests

S5 axioms: valid as expected
Barcan formulas: not valid for ’varying-dom.’ quant.
Barcan formulas: valid for ‘constant-dom.’ quant.

Hilbert axioms and rules: valid as expected

Existential import: yes, for ‘constant-domain’ quant.

Existential import: no, for ‘varying-domain’ quant.

```
theory TestsHOML imports HOMLinHOL
begin
—‹Test for S5 modal logic›
  lemma axM: "⌊□φ ⊃ φ⌋"                    using Rrefl by blast
  lemma axD: "⌊□φ ⊃ ◇φ⌋"                   using Rrefl by blast
  lemma axB: "⌊φ ⊃ □◇φ⌋"                   using Rsymm by blast
  lemma ax4: "⌊□φ ⊃ □□φ⌋"                  using Rtrans by blast
  lemma ax5: "⌊◇φ ⊃ □◇φ⌋"          using Rsymm Rtrans by blast
—‹Test for Barcan and converse Barcan formula:›
  lemma BarcanAct:     "⌊(∀ᴱx.□(φ x)) ⊃ □(∀ᴱx.(φ x))⌋"   nitpick oops —‹Countermodel found›
  lemma ConvBarcanAct: "⌊□(∀ᴱx.(φ x)) ⊃ (∀ᴱx.□(φ x))⌋"   nitpick oops —‹Countermodel found›
  lemma BarcanPoss:    "⌊(∀x.□(φ x)) ⊃ □(∀x. φ x)⌋"     by blast
  lemma ConvBarcanPoss: "⌊□(∀x.(φ x)) ⊃ (∀x.□(φ x))⌋"    by blast
—‹A simple Hilbert system for classical propositional logic is derived›
  lemma Hilbert_A1: "⌊A ⊃ (B ⊃ A)⌋"                                  by blast
  lemma Hilbert_A2: "⌊(A ⊃ (B ⊃ C)) ⊃ ((A ⊃ B) ⊃ (A ⊃ C))⌋"          by blast
  lemma Hilbert_MP: assumes "⌊A⌋" and "⌊A ⊃ B⌋" shows "⌊B⌋"  using assms by blast
—‹We have a polymorphic possibilist quantifier for which existential import holds›
  lemma Quant_1: assumes "⌊A⌋" shows "⌊∀x::'a. A⌋"   using assms by auto
—‹Existential import holds for possibilist quantifiers›
  lemma EximPossibilist1: "⌊∃x::e. x = x⌋"                          by blast
  lemma EximPossibilist2: "⌊∃x::e. x ≡ x⌋"                          by blast
  lemma EximPossibilist3: "⌊∃x::e. x = t⌋"                          by blast
  lemma EximPossibilist4: "⌊∃x::'a. x ≡ t::'a⌋"                     by blast
  lemma EximPossibilist:  "⌊∃x::'a. T⌋"                             by blast
—‹We have an actualist quantifier for individuals for which existential import does not hold›
  lemma Quant_2: assumes "⌊A⌋" shows "⌊∀ᴱx::e. A⌋" using assms by auto
—‹Existential import does not hold for our actualist quantifiers (for individuals)›
  lemma EximActualist1: "⌊∃ᴱx::e. x = x⌋"      nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist2: "⌊∃ᴱx::e. x ≡ x⌋"      nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist3: "⌊∃ᴱx::e. x = t⌋"      nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist:  "⌊∃ᴱx::e. T⌋"          nitpick[card=1] oops —‹Countermodel found›
```

Fig. 4 Tests and verifications of properties for the embedding of HOML in HOL from Fig. 3

HOML in HOL: tests

S5 axioms: valid as expected
Barcan formulas: not valid for ’varying-dom.’ quant.
Barcan formulas: valid for ‘constant-dom.’ quant.

Hilbert axioms and rules: valid as expected

Existential import: yes, for ‘constant-domain’ quant.

Existential import: no, for ‘varying-domain’ quant.

Properties of equality: valid as expected
(except for Boolean extensionality)

```
theory TestsHOML imports HOMLinHOL
begin
—‹Test for S5 modal logic›
  lemma axM: "⌊□φ ⊃ φ⌋"                    using Rrefl by blast
  lemma axD: "⌊□φ ⊃ ◇φ⌋"                   using Rrefl by blast
  lemma axB: "⌊φ ⊃ □◇φ⌋"                   using Rsymm by blast
  lemma ax4: "⌊□φ ⊃ □□φ⌋"                  using Rtrans by blast
  lemma ax5: "⌊◇φ ⊃ □◇φ⌋"          using Rsymm Rtrans by blast
—‹Test for Barcan and converse Barcan formula:›
  lemma BarcanAct:    "⌊(∀ᴱx.□(φ x)) ⊃ □(∀ᴱx.(φ x))⌋"   nitpick oops —‹Countermodel found›
  lemma ConvBarcanAct: "⌊□(∀ᴱx.(φ x)) ⊃ (∀ᴱx.□(φ x))⌋"  nitpick oops —‹Countermodel found›
  lemma BarcanPoss:   "⌊(∀x.□(φ x)) ⊃ □(∀x. φ x)⌋"     by blast
  lemma ConvBarcanPoss: "⌊□(∀x.(φ x)) ⊃ (∀x.□(φ x))⌋"  by blast
—‹A simple Hilbert system for classical propositional logic is derived›
  lemma Hilbert_A1: "⌊A ⊃ (B ⊃ A)⌋"                    by blast
  lemma Hilbert_A2: "⌊(A ⊃ (B ⊃ C)) ⊃ ((A ⊃ B) ⊃ (A ⊃ C))⌋"  by blast
  lemma Hilbert_MP: assumes "⌊A⌋" and "⌊A ⊃ B⌋" shows "⌊B⌋"  using assms by blast
—‹We have a polymorphic possibilist quantifier for which existential import holds›
  lemma Quant_1: assumes "⌊A⌋" shows "⌊∀x::'a. A⌋"   using assms by auto
—‹Existential import holds for possibilist quantifiers›
  lemma EximPossibilist1: "⌊∃x::e. x = x⌋"          by blast
  lemma EximPossibilist2: "⌊∃x::e. x ≡ x⌋"          by blast
  lemma EximPossibilist3: "⌊∃x::e. x = t⌋"          by blast
  lemma EximPossibilist4: "⌊∃x::'a. x ≡ t::'a⌋"     by blast
  lemma EximPossibilist:  "⌊∃x::'a. T⌋"             by blast
—‹We have an actualist quantifier for individuals for which existential import does not hold›
  lemma Quant_2: assumes "⌊A⌋" shows "⌊∀ᴱx::e. A⌋" using assms by auto
—‹Existential import does not hold for our actualist quantifiers (for individuals)›
  lemma EximActualist1: "⌊∃ᴱx::e. x = x⌋"    nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist2: "⌊∃ᴱx::e. x ≡ x⌋"    nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist3: "⌊∃ᴱx::e. x = t⌋"    nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist:  "⌊∃ᴱx::e. T⌋"        nitpick[card=1] oops —‹Countermodel found›
—‹Properties of the embedded primitive equality, which coincides with Leibniz equality›
  lemma EqRefl:    "⌊x = x⌋"                        by blast
  lemma EqSym:     "⌊(x = y) ↔ (y = x)⌋"            by blast
  lemma EqTrans:   "⌊((x = y) ∧ (y = z)) ⊃ (x = z)⌋"  by blast
  lemma EQCong:    "⌊(x = y) ⊃ ((φ x) = (φ y))⌋"    by blast
  lemma EQFuncExt: "⌊(φ = ψ) ⊃ (∀x. ((φ x) = (ψ x)))⌋"  by blast
  lemma EQBoolExt1: "⌊(φ = ψ) ⊃ (φ ↔ ψ)⌋"          by blast
  lemma EQBoolExt2: "⌊(φ ↔ ψ) ⊃ (φ = ψ)⌋"   nitpick[card=2] oops —‹Countermodel found›
  lemma EQBoolExt3: "⌊(φ ↔ ψ)⌋ ⟶ ⌊(φ = ψ)⌋"          by blast
  lemma EqPrimLeib: "⌊(x = y) ↔ (x ≡ y)⌋"           by auto
```

Fig. 4 Tests and verifications of properties for the embedding of HOML in HOL from Fig. 3

HOML in HOL: tests

S5 axioms: valid as expected
Barcan formulas: not valid for ’varying-dom.’ quant.
Barcan formulas: valid for ‘constant-dom.’ quant.

Hilbert axioms and rules: valid as expected

Existential import: yes, for ‘constant-domain’ quant.

Existential import: no, for ‘varying-domain’ quant.

Properties of equality: valid as expected
(except for Boolean extensionality)

Full comprehension: valid as expected

```
theory TestsHOML imports HOMLinHOL
begin
—‹Test for S5 modal logic›
  lemma axM: "⌊□φ ⊃ φ⌋"                    using Rrefl by blast
  lemma axD: "⌊□φ ⊃ ◇φ⌋"                   using Rrefl by blast
  lemma axB: "⌊φ ⊃ □◇φ⌋"                   using Rsymm by blast
  lemma ax4: "⌊□φ ⊃ □□φ⌋"                  using Rtrans by blast
  lemma ax5: "⌊◇φ ⊃ □◇φ⌋"          using Rsymm Rtrans by blast
—‹Test for Barcan and converse Barcan formula:›
  lemma BarcanAct:    "⌊(∀ᴱx.□(φ x)) ⊃ □(∀ᴱx.(φ x))⌋"   nitpick oops —‹Countermodel found›
  lemma ConvBarcanAct: "⌊□(∀ᴱx.(φ x)) ⊃ (∀ᴱx.□(φ x))⌋"  nitpick oops —‹Countermodel found›
  lemma BarcanPoss:   "⌊(∀x.□(φ x)) ⊃ □(∀x. φ x)⌋"     by blast
  lemma ConvBarcanPoss: "⌊□(∀x.(φ x)) ⊃ (∀x.□(φ x))⌋"  by blast
—‹A simple Hilbert system for classical propositional logic is derived›
  lemma Hilbert_A1: "⌊A ⊃ (B ⊃ A)⌋"                    by blast
  lemma Hilbert_A2: "⌊(A ⊃ (B ⊃ C)) ⊃ ((A ⊃ B) ⊃ (A ⊃ C))⌋"  by blast
  lemma Hilbert_MP: assumes "⌊A⌋" and "⌊A ⊃ B⌋" shows "⌊B⌋"  using assms by blast
—‹We have a polymorphic possibilist quantifier for which existential import holds›
  lemma Quant_1: assumes "⌊A⌋" shows "⌊∀x::'a. A⌋"   using assms by auto
—‹Existential import holds for possibilist quantifiers›
  lemma EximPossibilist1: "⌊∃x::e. x = x⌋"          by blast
  lemma EximPossibilist2: "⌊∃x::e. x ≡ x⌋"          by blast
  lemma EximPossibilist3: "⌊∃x::e. x = t⌋"          by blast
  lemma EximPossibilist4: "⌊∃x::'a. x ≡ t::'a⌋"     by blast
  lemma EximPossibilist:  "⌊∃x::'a. T⌋"             by blast
—‹We have an actualist quantifier for individuals for which existential import does not hold›
  lemma Quant_2: assumes "⌊A⌋" shows "⌊∀ᴱx::e. A⌋" using assms by auto
—‹Existential import does not hold for our actualist quantifiers (for individuals)›
  lemma EximActualist1: "⌊∃ᴱx::e. x = x⌋"    nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist2: "⌊∃ᴱx::e. x ≡ x⌋"    nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist3: "⌊∃ᴱx::e. x = t⌋"    nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist:  "⌊∃ᴱx::e. T⌋"        nitpick[card=1] oops —‹Countermodel found›
—‹Properties of the embedded primitive equality, which coincides with Leibniz equality›
  lemma EqRefl:     "⌊x = x⌋"                       by blast
  lemma EqSym:      "⌊(x = y) ↔ (y = x)⌋"           by blast
  lemma EqTrans:    "⌊((x = y) ∧ (y = z)) ⊃ (x = z)⌋"  by blast
  lemma EQCong:     "⌊(x = y) ⊃ ((φ x) = (φ y))⌋"   by blast
  lemma EQFuncExt:  "⌊(φ = ψ) ⊃ (∀x. ((φ x) = (ψ x)))⌋"  by blast
  lemma EQBoolExt1: "⌊(φ = ψ) ⊃ (φ ↔ ψ)⌋"          by blast
  lemma EQBoolExt2: "⌊(φ ↔ ψ) ⊃ (φ = ψ)⌋"    nitpick[card=2] oops —‹Countermodel found›
  lemma EQBoolExt3: "⌊(φ ↔ ψ)⌋ ⟶ ⌊(φ = ψ)⌋"          by blast
  lemma EqPrimLeib: "⌊(x = y) ↔ (x ≡ y)⌋"           by auto
—‹Comprehension is natively supported in HOL (due to λ-abstraction)›
  lemma Comprehension1: "⌊∃φ. ∀x. (φ x) ↔ A⌋"       by force
  lemma Comprehension2: "⌊∃φ. ∀x. (φ x) ↔ (A x)⌋"   by force
  lemma Comprehension3: "⌊∃φ. ∀x y. (φ x y) ↔ (A x y)⌋"  by force
```

Fig. 4 Tests and verifications of properties for the embedding of HOML in HOL from Fig. 3

HOML in HOL: tests

S5 axioms: valid as expected
Barcan formulas: not valid for ’varying-dom.’ quant.
Barcan formulas: valid for ‘constant-dom.’ quant.

Hilbert axioms and rules: valid as expected

Existential import: yes, for ‘constant-domain’ quant.

Existential import: no, for ‘varying-domain’ quant.

Properties of equality: valid as expected
(except for Boolean extensionality)

Full comprehension: valid as expected

Modal collapse: not valid

```
theory TestsHOML imports HOMLinHOL
begin
—‹Test for S5 modal logic›
  lemma axM: "⌊□φ ⊃ φ⌋"                    using Rrefl by blast
  lemma axD: "⌊□φ ⊃ ◇φ⌋"                   using Rrefl by blast
  lemma axB: "⌊φ ⊃ □◇φ⌋"                   using Rsymm by blast
  lemma ax4: "⌊□φ ⊃ □□φ⌋"                  using Rtrans by blast
  lemma ax5: "⌊◇φ ⊃ □◇φ⌋"          using Rsymm Rtrans by blast
—‹Test for Barcan and converse Barcan formula:›
  lemma BarcanAct:    "⌊(∀ᴱx.□(φ x)) ⊃ □(∀ᴱx.(φ x))⌋"   nitpick oops —‹Countermodel found›
  lemma ConvBarcanAct: "⌊□(∀ᴱx.(φ x)) ⊃ (∀ᴱx.□(φ x))⌋"  nitpick oops —‹Countermodel found›
  lemma BarcanPoss:   "⌊(∀x.□(φ x)) ⊃ □(∀x. φ x)⌋"      by blast
  lemma ConvBarcanPoss: "⌊□(∀x.(φ x)) ⊃ (∀x.□(φ x))⌋"   by blast
—‹A simple Hilbert system for classical propositional logic is derived›
  lemma Hilbert_A1: "⌊A ⊃ (B ⊃ A)⌋"                     by blast
  lemma Hilbert_A2: "⌊(A ⊃ (B ⊃ C)) ⊃ ((A ⊃ B) ⊃ (A ⊃ C))⌋"  by blast
  lemma Hilbert_MP: assumes "⌊A⌋" and "⌊A ⊃ B⌋" shows "⌊B⌋"  using assms by blast
—‹We have a polymorphic possibilist quantifier for which existential import holds›
  lemma Quant_1: assumes "⌊A⌋" shows "⌊∀x::'a. A⌋"   using assms by auto
—‹Existential import holds for possibilist quantifiers›
  lemma EximPossibilist1: "⌊∃x::e. x = x⌋"          by blast
  lemma EximPossibilist2: "⌊∃x::e. x ≡ x⌋"          by blast
  lemma EximPossibilist3: "⌊∃x::e. x = t⌋"          by blast
  lemma EximPossibilist4: "⌊∃x::'a. x ≡ t::'a⌋"     by blast
  lemma EximPossibilist:  "⌊∃x::'a. T⌋"             by blast
—‹We have an actualist quantifier for individuals for which existential import does not hold›
  lemma Quant_2: assumes "⌊A⌋" shows "⌊∀ᴱx::e. A⌋" using assms by auto
—‹Existential import does not hold for our actualist quantifiers (for individuals)›
  lemma EximActualist1: "⌊∃ᴱx::e. x = x⌋"   nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist2: "⌊∃ᴱx::e. x ≡ x⌋"   nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist3: "⌊∃ᴱx::e. x = t⌋"   nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist:  "⌊∃ᴱx::e. T⌋"       nitpick[card=1] oops —‹Countermodel found›
—‹Properties of the embedded primitive equality, which coincides with Leibniz equality›
  lemma EqRefl:    "⌊x = x⌋"                        by blast
  lemma EqSym:     "⌊(x = y) ↔ (y = x)⌋"            by blast
  lemma EqTrans:   "⌊((x = y) ∧ (y = z)) ⊃ (x = z)⌋"  by blast
  lemma EQCong:    "⌊(x = y) ⊃ ((φ x) = (φ y))⌋"    by blast
  lemma EQFuncExt: "⌊(φ = ψ) ⊃ (∀x. ((φ x) = (ψ x)))⌋"  by blast
  lemma EQBoolExt1: "⌊(φ = ψ) ⊃ (φ ↔ ψ)⌋"          by blast
  lemma EQBoolExt2: "⌊(φ ↔ ψ) ⊃ (φ = ψ)⌋"   nitpick[card=2] oops —‹Countermodel found›
  lemma EQBoolExt3: "⌊(φ ↔ ψ)⌋ ⟶ ⌊(φ = ψ)⌋"          by blast
  lemma EqPrimLeib: "⌊(x = y) ↔ (x ≡ y)⌋"           by auto
—‹Comprehension is natively supported in HOL (due to λ-abstraction)›
  lemma Comprehension1: "⌊∃φ. ∀x. (φ x) ↔ A⌋"       by force
  lemma Comprehension2: "⌊∃φ. ∀x. (φ x) ↔ (A x)⌋"   by force
  lemma Comprehension3: "⌊∃φ. ∀x y. (φ x y) ↔ (A x y)⌋"  by force
—‹Modal collapse does not hold›
  lemma ModalCollapse: "⌊∀φ. φ ⊃ □φ⌋"   nitpick[card=2] oops —‹Countermodel found›
```

Fig. 4 Tests and verifications of properties for the embedding of HOML in HOL from Fig. 3

HOML in HOL: tests

S5 axioms: valid as expected
Barcan formulas: not valid for ’varying-dom.’ quant.
Barcan formulas: valid for ‘constant-dom.’ quant.

Hilbert axioms and rules: valid as expected

Existential import: yes, for ‘constant-domain’ quant.

Existential import: no, for ‘varying-domain’ quant.

Properties of equality: valid as expected
(except for Boolean extensionality)

Full comprehension: valid as expected

Modal collapse: not valid

universal property = self-identity
Absurd/empty property = self-difference

```
theory TestsHOML imports HOMLinHOL
begin
—‹Test for S5 modal logic›
  lemma axM: "⌊□φ ⊃ φ⌋"                    using Rrefl by blast
  lemma axD: "⌊□φ ⊃ ◇φ⌋"                   using Rrefl by blast
  lemma axB: "⌊φ ⊃ □◇φ⌋"                   using Rsymm by blast
  lemma ax4: "⌊□φ ⊃ □□φ⌋"                  using Rtrans by blast
  lemma ax5: "⌊◇φ ⊃ □◇φ⌋"          using Rsymm Rtrans by blast
—‹Test for Barcan and converse Barcan formula:›
  lemma BarcanAct:     "⌊(∀ᴱx.□(φ x)) ⊃ □(∀ᴱx.(φ x))⌋"   nitpick oops —‹Countermodel found›
  lemma ConvBarcanAct: "⌊□(∀ᴱx.(φ x)) ⊃ (∀ᴱx.□(φ x))⌋"   nitpick oops —‹Countermodel found›
  lemma BarcanPoss:    "⌊(∀x.□(φ x)) ⊃ □(∀x. φ x)⌋"     by blast
  lemma ConvBarcanPoss: "⌊□(∀x.(φ x)) ⊃ (∀x.□(φ x))⌋"    by blast
—‹A simple Hilbert system for classical propositional logic is derived›
  lemma Hilbert_A1: "⌊A ⊃ (B ⊃ A)⌋"                      by blast
  lemma Hilbert_A2: "⌊(A ⊃ (B ⊃ C)) ⊃ ((A ⊃ B) ⊃ (A ⊃ C))⌋"   by blast
  lemma Hilbert_MP: assumes "⌊A⌋" and "⌊A ⊃ B⌋" shows "⌊B⌋"  using assms by blast
—‹We have a polymorphic possibilist quantifier for which existential import holds›
  lemma Quant_1: assumes "⌊A⌋" shows "⌊∀x::'a. A⌋"   using assms by auto
—‹Existential import holds for possibilist quantifiers›
  lemma EximPossibilist1: "⌊∃x::e. x = x⌋"                by blast
  lemma EximPossibilist2: "⌊∃x::e. x ≡ x⌋"                by blast
  lemma EximPossibilist3: "⌊∃x::e. x = t⌋"                by blast
  lemma EximPossibilist4: "⌊∃x::'a. x ≡ t::'a⌋"           by blast
  lemma EximPossibilist:  "⌊∃x::'a. T⌋"                   by blast
—‹We have an actualist quantifier for individuals for which existential import does not hold›
  lemma Quant_2: assumes "⌊A⌋" shows "⌊∀ᴱx::e. A⌋" using assms by auto
—‹Existential import does not hold for our actualist quantifiers (for individuals)›
  lemma EximActualist1: "⌊∃ᴱx::e. x = x⌋"     nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist2: "⌊∃ᴱx::e. x ≡ x⌋"     nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist3: "⌊∃ᴱx::e. x = t⌋"     nitpick[card=1] oops —‹Countermodel found›
  lemma EximActualist:  "⌊∃ᴱx::e. T⌋"         nitpick[card=1] oops —‹Countermodel found›
—‹Properties of the embedded primitive equality, which coincides with Leibniz equality›
  lemma EqRefl:      "⌊x = x⌋"                          by blast
  lemma EqSym:       "⌊(x = y) ↔ (y = x)⌋"              by blast
  lemma EqTrans:     "⌊((x = y) ∧ (y = z)) ⊃ (x = z)⌋"  by blast
  lemma EQCong:      "⌊(x = y) ⊃ ((φ x) = (φ y))⌋"      by blast
  lemma EQFuncExt:   "⌊(φ = ψ) ⊃ (∀x. ((φ x) = (ψ x)))⌋"  by blast
  lemma EQBoolExt1:  "⌊(φ = ψ) ⊃ (φ ↔ ψ)⌋"             by blast
  lemma EQBoolExt2:  "⌊(φ ↔ ψ) ⊃ (φ = ψ)⌋"   nitpick[card=2] oops —‹Countermodel found›
  lemma EQBoolExt3:  "⌊(φ ↔ ψ)⌋ ⟶ ⌊(φ = ψ)⌋"            by blast
  lemma EqPrimLeib:  "⌊(x = y) ↔ (x ≡ y)⌋"              by auto
—‹Comprehension is natively supported in HOL (due to λ-abstraction)›
  lemma Comprehension1: "⌊∃φ. ∀x. (φ x) ↔ A⌋"           by force
  lemma Comprehension2: "⌊∃φ. ∀x. (φ x) ↔ (A x)⌋"       by force
  lemma Comprehension3: "⌊∃φ. ∀x y. (φ x y) ↔ (A x y)⌋"  by force
—‹Modal collapse does not hold›
  lemma ModalCollapse: "⌊∀φ. φ ⊃ □φ⌋"   nitpick[card=2] oops —‹Countermodel found›
—‹Empty property and self-difference›
  lemma TruePropertyAndSelfIdentity:      "⌊(λx::e. T) = (λx. x = x)⌋"  by blast
  lemma EmptyPropertyAndSelfDifference:   "⌊(λx::e. ⊥) = (λx. x ≠ x)⌋"  by blast
  lemma EmptyProperty2: "⌊∃x. φ x⌋ ⟶ ⌊φ ≠ (λx::e. ⊥)⌋"                 by blast
  lemma EmptyProperty3: "⌊∃ᴱx. φ x⌋ ⟶ ⌊φ ≠ (λx::e. ⊥)⌋"                by blast


end
```

Fig. 4 Tests and verifications of properties for the embedding of HOML in HOL from Fig. 3

HOML in HOL: modal (ultra)filter

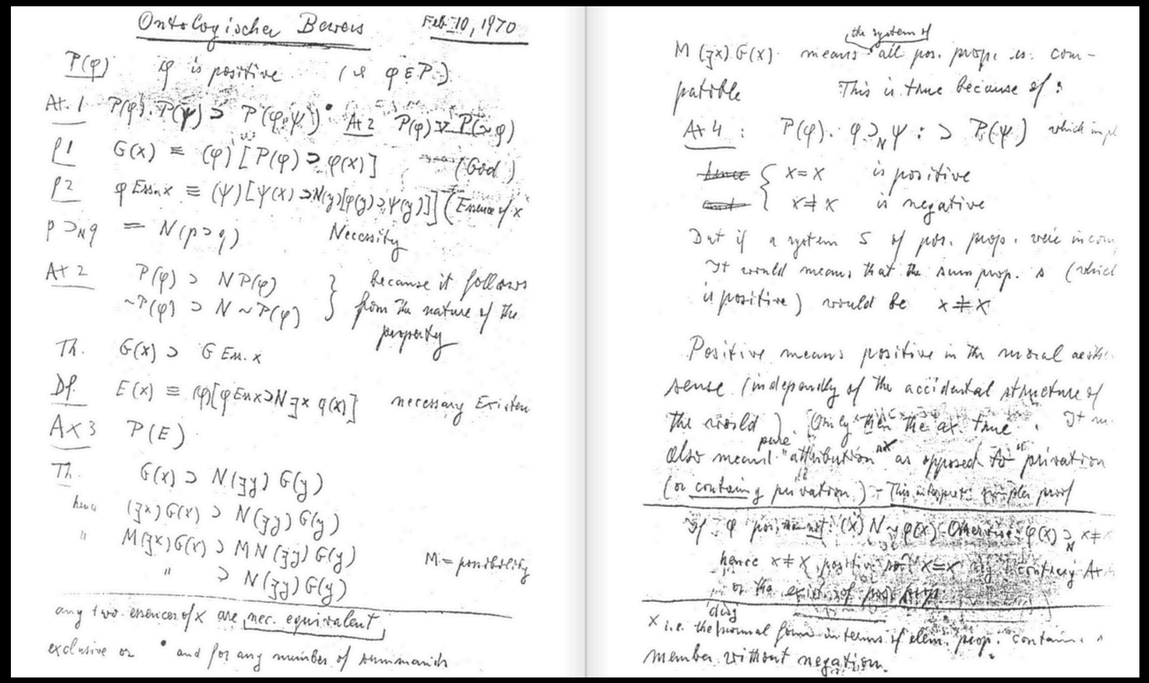
Set operations lifted to the HOML level

```
theory ModalFilter imports HOMLinHOL
begin
 type_synonym τ="e⇒σ"
 abbreviation Element::"τ⇒(τ⇒σ)⇒σ" (infix "∈" 90) where "φ∈S ≡ S φ"
 abbreviation EmptySet::τ ("∅") where "∅ ≡ λx. ⊥"
 abbreviation UniversalSet::τ ("U") where "U ≡ λx. ⊤"
 abbreviation Subset::"τ⇒τ⇒σ" (infix "⊆" 80) where "φ⊆ψ ≡ ∀x.((φ x) ⊃ (ψ x))"
 abbreviation SubsetE::"τ⇒τ⇒σ" (infix "⊆ᴱ" 80) where "φ⊆ᴱψ ≡ ∀ᴱx.((φ x) ⊃ (ψ x))"
 abbreviation Intersection::"τ⇒τ⇒τ" (infix "⊓" 91) where "φ⊓ψ ≡ λx.((φ x) ∧ (ψ x))"
 abbreviation Inverse::"τ⇒τ" ("⁻¹") where "⁻¹ψ ≡ λx. ¬(ψ x)"
 abbreviation "Filter Φ    ≡ U∈Φ ∧ ¬(∅∈Φ) ∧ (∀φ ψ. φ∈Φ ∧ φ⊆ᴱψ ⊃ ψ∈Φ) ∧ (∀φ ψ. φ∈Φ ∧ ψ∈Φ ⊃ φ⊓ψ∈Φ)"
 abbreviation "UFilter Φ   ≡ Filter Φ ∧ (∀φ. φ∈Φ ∨ (⁻¹φ)∈Φ)"
 abbreviation "FilterP Φ   ≡ U∈Φ ∧ ¬(∅∈Φ) ∧ (∀φ ψ. φ∈Φ ∧ φ⊆ψ ⊃ ψ∈Φ) ∧ (∀φ ψ. φ∈Φ ∧ ψ∈Φ ⊃ φ⊓ψ∈Φ)"
 abbreviation "UFilterP Φ ≡ FilterP Φ ∧ (∀φ. φ∈Φ ∨ (⁻¹φ)∈Φ)"
end
```

Fig. 5 Set filter and ultrafilter formalized for our modal logic setting

Notions of filter and ultrafilter lifted to the HOML level (for varying-/constant-dom. quant.)

- contains the universal set, but not the empty set
- closed under supersets and intersection
- maximal (only for ultrafilters)



Ontologischer Beweis Feb 10, 1970

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

Ax. 1 $P(\varphi) \cdot P(\psi) \supset P(\varphi \cdot \psi)$ · Ax 2 $P(\varphi) \vee P(\neg\varphi)$

D1 $G(x) \equiv (\varphi)[P(\varphi) \supset \varphi(x)]$ (God)

D2 $\varphi \, Ess.x \equiv (\psi)[\psi(x) \supset N(y)[\varphi(y) \supset \psi(y)]]$ (Essence of x)

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

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

Th. $G(x) \supset G \, Ess. x$

Df. $E(x) \equiv (\varphi)[\varphi \, Ess.x \supset N(\exists x) \varphi(x)]$ 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 mec. equivalent

exclusive or · and for any number of summands

M(∃x)G(x) means the system of all pos. propp. is compatible This is true because of:

Ax 4: $P(\varphi) \cdot \varphi \supset_N \psi : \supset P(\psi)$ which implies

~~three~~ $x = x$ is positive
~~not~~ $x \neq x$ is negative

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

Positive means: positive in the moral aesthetic sense (independently of the accidental structure of the world). Only then the ax. true. It must also mean: "attribution" as opposed to "privation" (or containing privation) - This interprets positive proof

If φ positive: $(x) N \neg \varphi(x)$. Otherwise $\varphi(x) \supset_N x \neq$ hence $x \neq x$ positive, but $x = x$ by contrary Ax or the existence of pos. propp.

dag

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

Formalisation of Gödel's manuscript of 1970 (scriptum notion of essence: Gödel1)

— inconsistency —

Gödel 1

Positive properties: P

Axiom Ax1: $P\varphi \wedge P\psi \supset P(\varphi \,.\, \psi)$

Axiom Ax2a: $P\varphi \vee^{e} P{\sim}\varphi$

```
1 theory GoedelVariantHOML1 imports HOMLinHOL
2 begin
3   consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
4   axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ . ψ)⌋"
5   axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
```

P: Positive properties (intensions)

Ax1:
The conjunction of two positive properties is positive.

Ax2a:
Either a property or its complement is positive.

Fig. 6 Gödel's axioms and definitions, as presented in the 1970 manuscript, are inconsistent

Gödel 1

Definition of God-like: $G\,x \equiv \forall\varphi.\, P\varphi \supset \varphi\,x$

```
theory GoedelVariantHOML1 imports HOMLinHOL
begin
  consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
  axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ . ψ)⌋"
  axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
  definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
```

G(od-like):
An entity x is God-like
iff
it possesses all positive properties.

Fig. 6 Gödel's axioms and definitions, as presented in the 1970 manuscript, are inconsistent

Gödel 1

Definition of Essence: φ ***Ess.*** $x \equiv \forall \psi.\ \psi\ x \supset (\varphi \supset_N \psi)$

```
1 theory GoedelVariantHOML1 imports HOMLinHOL
2 begin
3   consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
4   axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ . ψ)⌋"
5   axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
6   definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
7   abbreviation PropertyInclusion ("_⊃ₙ_") where "φ ⊃ₙ ψ ≡ □(∀ᴱy. φ y ⊃ ψ y)"
8   definition Essence ("_Ess._") where "φ Ess. x ≡ ∀ψ. ψ x ⊃ (φ ⊃ₙ ψ)"
```

Ess(ence):

A property φ is an essence of x

iff

all properties of x are necessarily implied by φ.

Fig. 6 Gödel's axioms and definitions, as presented in the 1970 manuscript, are inconsistent

Gödel 1

Axiom Ax2b: $P\,\varphi \supset \Box P\,\varphi$ ***and*** $\neg P\,\varphi \supset \Box(\neg P\,\varphi)$

```
theory GoedelVariantHOML1 imports HOMLinHOL
begin
  consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
  axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ • ψ)⌋"
  axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
  definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
  abbreviation PropertyInclusion ("_⊃ₙ_") where "φ ⊃ₙ ψ ≡ □(∀ᴱy. φ y ⊃ ψ y)"
  definition Essence ("_Ess._") where "φ Ess. x ≡ ∀ψ. ψ x ⊃ (φ ⊃ₙ ψ)"
  axiomatization where Ax2b: "⌊P φ ⊃ □ P φ⌋"
```

Ax2b:

Positive properties are necessarily positive.

Fig. 6 Gödel's axioms and definitions, as presented in the 1970 manuscript, are inconsistent

Gödel 1

Axiom Ax2b: $P\,\varphi \supset \Box P\,\varphi$ ***and*** $\neg P\,\varphi \supset \Box(\neg P\,\varphi)$

```
theory GoedelVariantHOML1 imports HOMLinHOL
begin
  consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
  axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ • ψ)⌋"
  axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
  definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
  abbreviation PropertyInclusion ("_⊃ₙ_") where "φ ⊃ₙ ψ ≡ □(∀ᴱy. φ y ⊃ ψ y)"
  definition Essence ("_Ess._") where "φ Ess. x ≡ ∀ψ. ψ x ⊃ (φ ⊃ₙ ψ)"
  axiomatization where Ax2b: "⌊P φ ⊃ □ P φ⌋"
  lemma Ax2b': "⌊¬P φ ⊃ □(¬P φ)⌋" using Ax2a Ax2b by blast
```

Lemma:

Non-positive properties are necessarily non-positive.

Fig. 6 Gödel's axioms and definitions, as presented in the 1970 manuscript, are inconsistent

Gödel 1

Theorem Th1: $G\,x \supset G\,Ess.\,x$

```
theory GoedelVariantHOML1 imports HOMLinHOL
begin
  consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
  axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ • ψ)⌋"
  axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
  definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
  abbreviation PropertyInclusion ("_⊃ₙ_") where "φ ⊃ₙ ψ ≡ □(∀ᴱy. φ y ⊃ ψ y)"
  definition Essence ("_Ess._") where "φ Ess. x ≡ ∀ψ. ψ x ⊃ (φ ⊃ₙ ψ)"
  axiomatization where Ax2b: "⌊P φ ⊃ □ P φ⌋"
  lemma Ax2b': "⌊¬P φ ⊃ □(¬P φ)⌋" using Ax2a Ax2b by blast
  theorem Th1: "⌊G x ⊃ G Ess. x⌋" using Ax2a Ax2b Essence_def God_def by (smt (verit))
```

Theorem 1:
Being God-like is an essence of every God-like entity.

Fig. 6 Gödel's axioms and definitions, as presented in the 1970 manuscript, are inconsistent

Gödel 1

Definition of Necessary Existence: $E\,x \equiv \forall\varphi.\,\varphi\,Ess.\,x \supset \Box(\exists^E x.\,\varphi\,x)$

Axiom Ax3: $P\,E$

```
theory GoedelVariantHOML1 imports HOMLinHOL
begin
  consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
  axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ • ψ)⌋"
  axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
  definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
  abbreviation PropertyInclusion ("_⊃ₙ_") where "φ ⊃ₙ ψ  ≡  □(∀ᴱy.  φ y ⊃ ψ y)"
  definition Essence ("_Ess._") where "φ Ess. x ≡ ∀ψ. ψ x ⊃ (φ ⊃ₙ ψ)"
  axiomatization where Ax2b: "⌊P φ ⊃ □ P φ⌋"
  lemma Ax2b': "⌊¬P φ ⊃ □(¬P φ)⌋" using Ax2a Ax2b by blast
  theorem Th1: "⌊G x ⊃ G Ess. x⌋" using Ax2a Ax2b Essence_def God_def by (smt (verit))
  definition NecExist ("E") where "E x  ≡ ∀φ. φ Ess. x ⊃ □(∃ᴱx. φ x)"
  axiomatization where Ax3: "⌊P E⌋"
```

E(xistence):

x has the property of existence

iff

all essences of x are necessarily exemplified.

Ax3:

E(xistence) is a positive property.

Fig. 6 Gödel's axioms and definitions, as presented in the 1970 manuscript, are inconsistent

Gödel 1

$$\textit{Theorem Th2: } \boldsymbol{G}\,x \supset \Box(\exists^{E} y.\, \boldsymbol{G}\,y)$$

```
theory GoedelVariantHOML1 imports HOMLinHOL
begin
  consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
  axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ . ψ)⌋"
  axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
  definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
  abbreviation PropertyInclusion ("_⊃ₙ_") where "φ ⊃ₙ ψ  ≡  □(∀ᴱy.  φ y ⊃ ψ y)"
  definition Essence ("_Ess._") where "φ Ess. x ≡ ∀ψ. ψ x ⊃ (φ ⊃ₙ ψ)"
  axiomatization where Ax2b: "⌊P φ ⊃ □ P φ⌋"
  lemma Ax2b': "⌊¬P φ ⊃ □(¬P φ)⌋" using Ax2a Ax2b by blast
  theorem Th1: "⌊G x ⊃ G Ess. x⌋" using Ax2a Ax2b Essence_def God_def by (smt (verit))
  definition NecExist ("E") where "E x  ≡ ∀φ. φ Ess. x ⊃ □(∃ᴱx. φ x)"
  axiomatization where Ax3: "⌊P E⌋"
  theorem Th2: "⌊G x ⊃ □(∃ᴱy. G y)⌋" using Ax3 Th1 God_def NecExist_def by smt
```

Theorem 2:

If x is God-like, then the property of being God-like is necessarily exemplified.

Fig. 6 Gödel's axioms and definitions, as presented in the 1970 manuscript, are inconsistent

Gödel 1

$$\textit{Theorem Th3: } \Diamond(\exists^E x.\, G\, x) \supset \Box(\exists^E y.\, G\, y)$$

Logic KB (Rsymm) needed here

```
theory GoedelVariantHOML1 imports HOMLinHOL
begin
 consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
 axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ . ψ)⌋"
 axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
 definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
 abbreviation PropertyInclusion ("_⊃ₙ_") where "φ ⊃ₙ ψ ≡ □(∀ᴱy. φ y ⊃ ψ y)"
 definition Essence ("_Ess._") where "φ Ess. x ≡ ∀ψ. ψ x ⊃ (φ ⊃ₙ ψ)"
 axiomatization where Ax2b: "⌊P φ ⊃ □ P φ⌋"
 lemma Ax2b': "⌊¬P φ ⊃ □(¬P φ)⌋" using Ax2a Ax2b by blast
 theorem Th1: "⌊G x ⊃ G Ess. x⌋" using Ax2a Ax2b Essence_def God_def by (smt (verit))
 definition NecExist ("E") where "E x ≡ ∀φ. φ Ess. x ⊃ □(∃ᴱx. φ x)"
 axiomatization where Ax3: "⌊P E⌋"
 theorem Th2: "⌊G x ⊃ □(∃ᴱy. G y)⌋" using Ax3 Th1 God_def NecExist_def by smt
 theorem Th3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" sledgehammer(Th2 Rsymm) —‹Proof found›
   proof -
    have 1: "⌊(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using Th2 by blast
    have 2: "⌊◇(∃ᴱx. G x) ⊃ ◇□(∃ᴱy. G y)⌋" using 1 by blast
    have 3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using 2 Rsymm by blast
    thus ?thesis by blast
  qed
```

Theorem:

The possible existence of a God-like entity implies the necessary existence of a God-like entity.

Fig. 6 Gödel's axioms and definitions, as presented in the 1970 manuscript, are inconsistent

Gödel 1

$$\textit{Axiom Ax4: } \boldsymbol{P}\,\varphi \wedge (\varphi \supset_N \psi) \supset \boldsymbol{P}\,\psi$$

```
theory GoedelVariantHOML1 imports HOMLinHOL
begin
 consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
 axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ . ψ)⌋"
 axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
 definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
 abbreviation PropertyInclusion ("_⊃ₙ_") where "φ ⊃ₙ ψ  ≡  □(∀ᴱy.  φ y ⊃ ψ y)"
 definition Essence ("_Ess._") where "φ Ess. x ≡ ∀ψ. ψ x ⊃ (φ ⊃ₙ ψ)"
 axiomatization where Ax2b: "⌊P φ ⊃ □ P φ⌋"
 lemma Ax2b': "⌊¬P φ ⊃ □(¬P φ)⌋" using Ax2a Ax2b by blast
 theorem Th1: "⌊G x ⊃ G Ess. x⌋" using Ax2a Ax2b Essence_def God_def by (smt (verit))
 definition NecExist ("E") where "E x  ≡ ∀φ. φ Ess. x ⊃ □(∃ᴱx. φ x)"
 axiomatization where Ax3: "⌊P E⌋"
 theorem Th2: "⌊G x ⊃ □(∃ᴱy. G y)⌋" using Ax3 Th1 God_def NecExist_def by smt
 theorem Th3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" sledgehammer(Th2 Rsymm)  —‹Proof found›
   proof -
    have 1: "⌊(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using Th2 by blast
    have 2: "⌊◇(∃ᴱx. G x) ⊃ ◇□(∃ᴱy. G y)⌋" using 1 by blast
    have 3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using 2 Rsymm by blast
    thus ?thesis by blast
  qed
 axiomatization where Ax4: "⌊P φ ∧ (φ ⊃ₙ ψ) ⊃ P ψ⌋"
```

Ax4:

Properties that are necessarily implied by a positive property are positive.

Fig. 6 Gödel's axioms and definitions, as presented in the 1970 manuscript, are inconsistent

Gödel 1

No model found!

```
theory GoedelVariantHOML1 imports HOMLinHOL
begin
 consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
 axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ • ψ)⌋"
 axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
 definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
 abbreviation PropertyInclusion ("_⊃ₙ_") where "φ ⊃ₙ ψ  ≡  □(∀ᴱy.  φ y ⊃ ψ y)"
 definition Essence ("_Ess._") where "φ Ess. x ≡ ∀ψ. ψ x ⊃ (φ ⊃ₙ ψ)"
 axiomatization where Ax2b: "⌊P φ ⊃ □ P φ⌋"
 lemma Ax2b': "⌊¬P φ ⊃ □(¬P φ)⌋" using Ax2a Ax2b by blast
 theorem Th1: "⌊G x ⊃ G Ess. x⌋" using Ax2a Ax2b Essence_def God_def by (smt (verit))
 definition NecExist ("E") where "E x  ≡ ∀φ. φ Ess. x ⊃ □(∃ᴱx. φ x)"
 axiomatization where Ax3: "⌊P E⌋"
 theorem Th2: "⌊G x ⊃ □(∃ᴱy. G y)⌋" using Ax3 Th1 God_def NecExist_def by smt
 theorem Th3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" sledgehammer(Th2 Rsymm)  —‹Proof found›
   proof -
    have 1: "⌊(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using Th2 by blast
    have 2: "⌊◇(∃ᴱx. G x) ⊃ ◇□(∃ᴱy. G y)⌋" using 1 by blast
    have 3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using 2 Rsymm by blast
    thus ?thesis by blast
  qed
 axiomatization where Ax4: "⌊P φ ∧ (φ ⊃ₙ ψ) ⊃ P ψ⌋"
 lemma True nitpick[satisfy] oops  —‹No model found›
```

Consistency check with Nitpick:

… no model found …

Fig. 6 Gödel's axioms and definitions, as presented in the 1970 manuscript, are inconsistent

Gödel 1

Inconsistency detected by theorem provers

The inconsistency follows via the "empty essence lemma":

(λx.⊥) Ess. x

Adapt the notion of essence?

```
theory GoedelVariantHOML1 imports HOMLinHOL
begin
 consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
 axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ . ψ)⌋"
 axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
 definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
 abbreviation PropertyInclusion ("_⊃ₙ_") where "φ ⊃ₙ ψ ≡ □(∀ᴱy. φ y ⊃ ψ y)"
 definition Essence ("_Ess._") where "φ Ess. x ≡ ∀ψ. ψ x ⊃ (φ ⊃ₙ ψ)"
 axiomatization where Ax2b: "⌊P φ ⊃ □ P φ⌋"
 lemma Ax2b': "⌊¬P φ ⊃ □(¬P φ)⌋" using Ax2a Ax2b by blast
 theorem Th1: "⌊G x ⊃ G Ess. x⌋" using Ax2a Ax2b Essence_def God_def by (smt (verit))
 definition NecExist ("E") where "E x ≡ ∀φ. φ Ess. x ⊃ □(∃ᴱx. φ x)"
 axiomatization where Ax3: "⌊P E⌋"
 theorem Th2: "⌊G x ⊃ □(∃ᴱy. G y)⌋" using Ax3 Th1 God_def NecExist_def by smt
 theorem Th3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" sledgehammer(Th2 Rsymm) —‹Proof found›
   proof -
    have 1: "⌊(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using Th2 by blast
    have 2: "⌊◇(∃ᴱx. G x) ⊃ ◇□(∃ᴱy. G y)⌋" using 1 by blast
    have 3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using 2 Rsymm by blast
    thus ?thesis by blast
  qed
 axiomatization where Ax4: "⌊P φ ∧ (φ ⊃ₙ ψ) ⊃ P ψ⌋"
 lemma True nitpick[satisfy] oops —‹No model found›
 lemma EmptyEssL: "⌊(λy.⊥) Ess. x⌋" using Essence_def by auto
```

Fig. 6 Gödel's axioms and definitions, as presented in the 1970 manuscript, are inconsistent

Gödel 1

Inconsistency detected by theorem provers

The inconsistency follows via the "empty essence lemma":

(λx.⊥) Ess. x

Adapt the notion of essence?

```
theory GoedelVariantHOML1 imports HOMLinHOL
begin
 consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
 axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ . ψ)⌋"
 axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
 definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
 abbreviation PropertyInclusion ("_⊃ₙ_") where "φ ⊃ₙ ψ ≡ □(∀ᴱy. φ y ⊃ ψ y)"
 definition Essence ("_Ess._") where "φ Ess. x ≡ ∀ψ. ψ x ⊃ (φ ⊃ₙ ψ)"
 axiomatization where Ax2b: "⌊P φ ⊃ □ P φ⌋"
 lemma Ax2b': "⌊¬P φ ⊃ □(¬P φ)⌋" using Ax2a Ax2b by blast
 theorem Th1: "⌊G x ⊃ G Ess. x⌋" using Ax2a Ax2b Essence_def God_def by (smt (verit))
 definition NecExist ("E") where "E x ≡ ∀φ. φ Ess. x ⊃ □(∃ᴱx. φ x)"
 axiomatization where Ax3: "⌊P E⌋"
 theorem Th2: "⌊G x ⊃ □(∃ᴱy. G y)⌋" using Ax3 Th1 God_def NecExist_def by smt
 theorem Th3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" sledgehammer(Th2 Rsymm) —‹Proof found›
   proof -
    have 1: "⌊(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using Th2 by blast
    have 2: "⌊◇(∃ᴱx. G x) ⊃ ◇□(∃ᴱy. G y)⌋" using 1 by blast
    have 3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using 2 Rsymm by blast
    thus ?thesis by blast
   qed
 axiomatization where Ax4: "⌊P φ ∧ (φ ⊃ₙ ψ) ⊃ P ψ⌋"
 lemma True nitpick[satisfy] oops —‹No model found›
 lemma EmptyEssL: "⌊(λy.⊥) Ess. x⌋" using Essence_def by auto
 theorem Inconsistency: False sledgehammer(Ax2a Ax3 Ax4 EmptyEssL NecExist_def) —‹Proof found›
  proof -
    have 1: "⌊¬(P (λx.⊥))⌋" using Ax2a Ax4 by blast
    have 2: "⌊P (λx.(λy.⊥) Ess. x ⊃ □(∃ᴱz. (λy.⊥)z))⌋" using Ax3 Ax4 NecExist_def by smt
    have 3: "⌊P (λx.□(∃ᴱz. (λx.⊥) z))⌋" using 2 EmptyEssL by simp
    have 4: "⌊P (λx.□⊥)⌋" using 3 by auto
    have 5: "⌊P (λx.⊥)⌋" using 4 Ax2a Ax4 by smt
    have 6: "⌊⊥⌋" using 1 5 by blast
    thus ?thesis by blast
  qed
end
```

Inconsistency

… proved …

Fig. 6 Gödel's axioms and definitions, as presented in the 1970 manuscript, are inconsistent

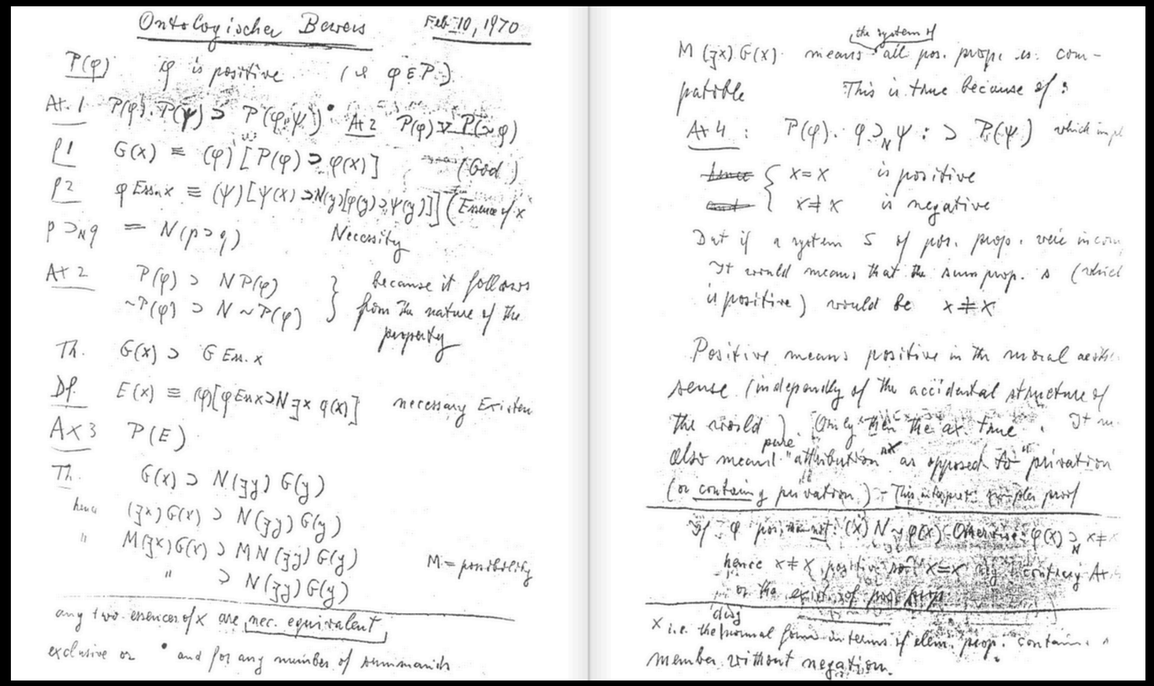
Gödel 1 — Results

(Positive) properties as intensional predicates

‘φ necessarily implies ψ’ defined as $\Box \forall^E x \,.\, \varphi x \supset \psi x$

Ax1: The conjunction of two positive properties is positive.
Ax2a: Either a property or its complement is positive.
G(od-like): x is God-like iff it possesses all positive properties.
Ess(ence): φ is an essence of x iff all properties of x are necessarily implied by φ.
Ax2b: Positive properties are necessarily positive.
Theorem 1: Being God-like is the essence of every God-like entity.
E(xistence): x has existence iff all essences of x are necessarily exemplified.
Ax3: E(xistence) is a positive property.
Theorem 2: If x is God-like, then God-likeness is necessarily exemplified.
Theorem 3: The possible existence of a God-like entity implies the necessary existence of a God-like entity.
Ax4: Properties that are necessarily implied by a positive property are positive.

These postulates are **inconsistent** (already in **modal logic K**)



Ontologischer Beweis Feb. 10, 1970

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

Ax. 1 $P(\varphi) \cdot P(\psi) \supset P(\varphi \psi)$ • Ax 2 $P(\varphi) \vee P(\sim\varphi)$

D1 $G(x) \equiv (\varphi)[P(\varphi) \supset \varphi(x)]$ (God)

D2 $\varphi \, Ess.x \equiv (\psi)[\psi(x) \supset N(y)[\varphi(y) \supset \psi(y)]]$ (Essence of x)

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

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

Th. $G(x) \supset G \, Ess. x$

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

Ax 3 $P(E)$

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

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

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

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

any two essences of x are mec. equivalent

exclusive or • and for any number of summands

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

Ax 4 : $P(\varphi) \cdot \varphi \supset_N \psi : \supset P(\psi)$ which im

~~Def~~ $x = x$ is positive

~~Def~~ $x \neq x$ is negative

Def if a system S of pos. propos. were inconsistent it would mean, that the sum prop. is (which is positive) would be $x \neq x$

Positive means: positive in the moral aesthetic sense (independently of the accidental structure of the world). Only then the at. time. It must also mean: "attribution" as opposed to privation (or containing privation) - This interpretation [illegible]

If φ privative: $(x) N \sim \varphi(x)$ Otherwise $\varphi(x) \supset_N x \neq$

hence $x \neq x$ positive [illegible] $x = x$ by contrary Ax.

[illegible] the existence [illegible]

dis

x i.e. the normal form in terms of elem. prop. contains a

member without negation.

Formalisation of Gödel's manuscript of 1970 (adapted notion of essence and generalized axiom Ax1Gen: Gödel2)

— proof in KB —

$\varphi \supset \Box\Diamond\varphi$

```
theory GoedelVariantHOML2 imports HOMLinHOL ModalFilter
begin
 consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
 axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ . ψ)⌋"
 axiomatization where Ax2a: "⌊P φ ∨ᵉ P ~φ⌋"
 definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
 abbreviation PropertyInclusion ("_⊃ₙ_") where "φ ⊃ₙ ψ ≡ □(∀ᴱy. φ y ⊃ ψ y)"
(**) definition Essence ("_Ess._") where "φ Ess. x ≡ φ x ∧ (∀ψ. ψ x ⊃ (φ ⊃ₙ ψ))"
 axiomatization where Ax2b: "⌊P φ ⊃ □ P φ⌋"
 lemma Ax2b': "⌊¬P φ ⊃ □(¬P φ)⌋" using Ax2a Ax2b by blast
 theorem Th1: "⌊G x ⊃ G Ess. x⌋" using Ax2a Ax2b Essence_def God_def by (smt (verit))
 definition NecExist ("E") where "E x  ≡ ∀φ. φ Ess. x ⊃ □(∃ᴱx. φ x)"
 axiomatization where Ax3: "⌊P E⌋"
 theorem Th2: "⌊G x ⊃ □(∃ᴱy. G y)⌋" using Ax3 Th1 God_def NecExist_def by smt
 theorem Th3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" sledgehammer(Th2 Rsymm) —‹Proof found›
   proof -
     have 1: "⌊(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using Th2 by blast
     have 2: "⌊◇(∃ᴱx. G x) ⊃ ◇□(∃ᴱy. G y)⌋" using 1 by blast
     have 3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using 2 Rsymm by blast
     thus ?thesis by blast
   qed
 axiomatization where Ax4: "⌊P φ ∧ (φ ⊃ₙ ψ) ⊃ P ψ⌋"
 lemma True nitpick[satisfy,card=1,eval="⌊P (λx. ⊥)⌋"] oops —‹One model found of cardinality one›
 abbreviation "PosProps Φ ≡ ∀φ. Φ φ ⊃ P φ"
 abbreviation "ConjOfPropsFrom φ Φ ≡ □(∀ᴱz. φ z ↔ (∀ψ. Φ ψ ⊃ ψ z))"
 axiomatization where Ax1Gen: "⌊(PosProps Φ ∧ ConjOfPropsFrom φ Φ) ⊃ P φ⌋"
 lemma L:      "⌊P G⌋"            using Ax1Gen God_def by smt
 theorem Th4: "⌊◇(∃ᴱx. G x)⌋"  using Ax2a Ax4 L by blast
 theorem Th5: "⌊□(∃ᴱx. G x)⌋"  using Th3 Th4 by blast
 lemma MC:    "⌊φ ⊃ □φ⌋"  sledgehammer(Ax2a Ax2b Th5 God_def Rsymm) —‹Proof found›
   proof - {fix w fix Q
     have 1: "∀x.(G x w ⟶ (∀Z. Z x ⊃ □(∀ᴱz. G z ⊃ Z z)) w)" using Ax2a Ax2b God_def by smt
     have 2: "(∃x. G x w)⟶((Q ⊃ □(∀ᴱz. G z ⊃ Q)) w)" using 1 by force
     have 3: "(Q ⊃ □Q) w" using 2 Th5 Rsymm by blast}
     thus ?thesis by auto
   qed
 lemma PosProps:     "⌊P (λx.⊤) ∧  P (λx. x = x)⌋"   using Ax2a Ax4 by blast
 lemma NegProps:     "⌊¬P(λx.⊥) ∧ ¬P(λx. x ≠ x)⌋"   using Ax2a Ax4 by blast
 lemma UniqueEss1: "⌊φ Ess. x ∧ ψ Ess. x ⊃  □(∀ᴱy. φ y ↔ ψ y)⌋" using Essence_def by smt
 lemma UniqueEss2: "⌊φ Ess. x ∧ ψ Ess. x ⊃  □(φ ≡ ψ)⌋" nitpick[card i=1] oops —‹Countermodel found›
 lemma UniqueEss3: "⌊φ Ess. x ⊃  □(∀ᴱy. φ y ⊃ y ≡ x)⌋" using Essence_def MC by auto
 lemma Monotheism: "⌊G x ∧ G y ⊃  x ≡ y⌋" using Ax2a God_def by smt
 lemma Filter:       "⌊Filter P⌋"     using Ax1 Ax4 MC NegProps PosProps Rsymm by smt
 lemma UltraFilter: "⌊UFilter P⌋"  using Ax2a Filter by smt
 lemma True nitpick[satisfy,card=1,eval="⌊P (λx.⊥)⌋"] oops —‹One model found of cardinality one›
end
```

Fig. 7 After an appropriate modification of the definition of essence in Gödel's 1970 ontological proof, the inconsistency revealed in Fig. 6 is avoided, and the argument can be successfully verified in modal logic S5 (indeed, as shown, only symmetry of the accessibility relation is actually needed)

Formalisation of Gödel's manuscript of 1970 (adapted notion of essence and generalized axiom Ax1Gen: Gödel2)

— proof in KB —

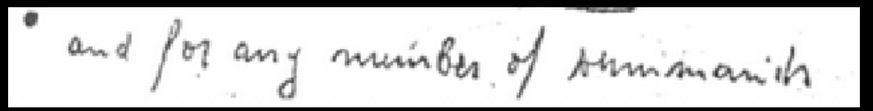
$\varphi \supset \Box\Diamond\varphi$

Gödel 2 — Results

‘φ necessarily implies ψ’ iff $\Box \forall^E x \,.\, \varphi x \supset \psi x$

Ax1Gen: The conjunction **of arbitrarily many** positive properties is positive.
Ax2a: Either a property or its complement is positive.
G(od-like): x is God-like iff it possesses all positive properties.
Ess(ence): φ is an essence of x iff all properties of x are necessarily implied by φ **and x has property φ**.
Ax2b: Positive properties are necessarily positive.
Theorem 1: Being God-like is the essence of every God-like entity.
E(xistence): x has existence iff all essences of x are necessarily exemplified.
Ax3: E(xistence) is a positive property.
Theorem 2: If x is God-like, then God-likeness is necessarily exemplified.
Theorem 3: The possible existence of a God-like entity implies the necessary existence of a God-like entity.
Ax4: Properties that are necessarily implied by a positive property are positive.
Lemma L: God-likeness is a positive property.
Theorem 4: A God-like entity possibly exists.
Theorem 5: A God-like entity necessarily exists.

Generalization of Ax1: **Ax1Gen**

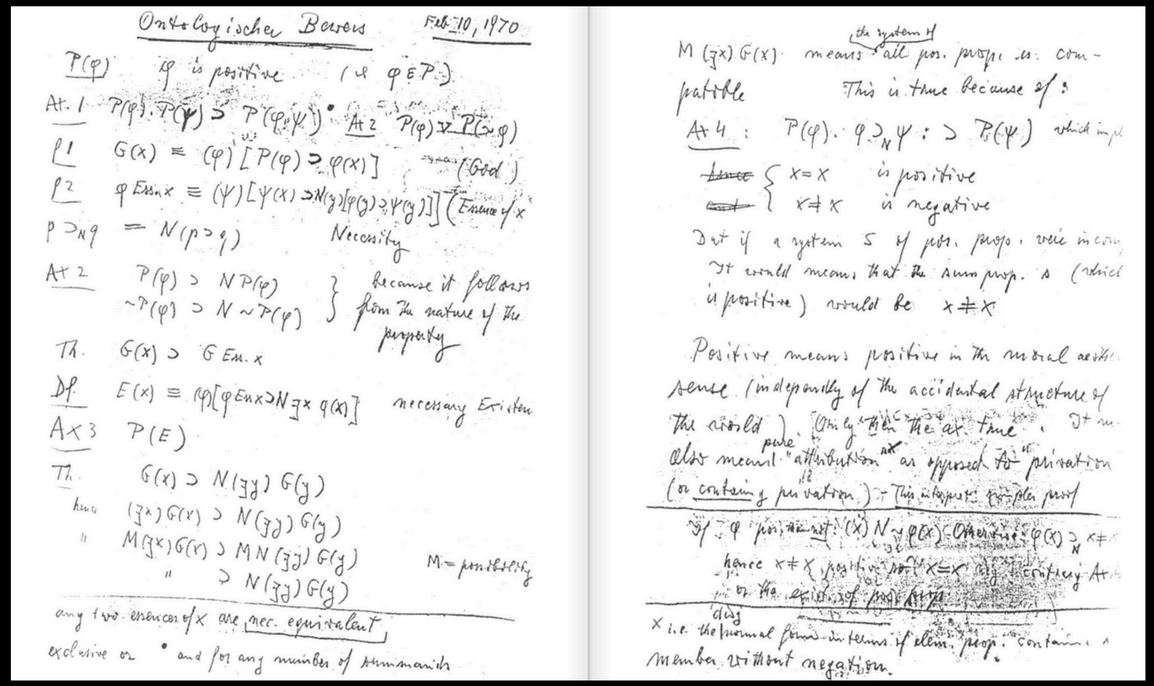

and for any number of humanity

Axiom Ax1Gen: $(PosProps\ \Phi \wedge ConjOfPropsFrom\ \varphi\ \Phi) \supset P\ \varphi$

```
abbreviation "PosProps Φ ≡ ∀φ. Φ φ ⊃ P φ"
abbreviation "ConjOfPropsFrom φ Φ ≡ □(∀ᴱz. φ z ↔ (∀ψ. Φ ψ ⊃ ψ z))"
axiomatization where Ax1Gen: "⌊(PosProps Φ ∧ ConjOfPropsFrom φ Φ) ⊃ P φ⌋"
```

Ax1Gen expressible in third-order logic

- Consistent
- Valid already in modal logic **KB**
- **Modal collapse** follows
- **Monotheism** follows
- **P** is an **ultrafilter**
- Self-identity: positive property
- Self-difference: non-positive prop.



Ontologischer Beweis Feb. 10, 1970

P(φ) φ is positive (i.e. φ ∈ P)

Ax. 1 P(φ) . P(ψ) ⊃ P(φ ∧ ψ) · Ax 2 P(φ) ∨ P(¬φ)

D1 G(x) ≡ (φ)[P(φ) ⊃ φ(x)] "x is God"

D2 φ Ess. x ≡ (ψ)[ψ(x) ⊃ N(y)[φ(y) ⊃ ψ(y)]] "Essence of x"

p ⊃_N q = N(p ⊃ q) Necessity

Ax 2 P(φ) ⊃ N P(φ)
¬P(φ) ⊃ N ¬P(φ) } because it follows from the nature of the property

Th. G(x) ⊃ G Ess. x

Df E(x) ≡ (φ)[φ Ess. x ⊃ N ∃x φ(x)] necessary Existenz

Ax 3 P(E)

Th. G(x) ⊃ N(∃y) G(y)

hence (∃x) G(x) ⊃ N(∃y) G(y)

" M(∃x) G(x) ⊃ M N(∃y) G(y) M = possibility

" ⊃ N(∃y) G(y)

any two essences of x are mec. equivalent

exclusive or · and for any number of summands

M(∃x) G(x) means: the system H all pos. propos. is consistible This is true because of:

Ax 4 : P(φ) . φ ⊃_N ψ : ⊃ P(ψ) which implies

~~three~~ { x = x is positive
~~not~~ { x ≠ x is negative

Def if a system S of pos. propos. were inconsistent it would mean, that the sum prop. of (which is positive) would be x ≠ x

Positive means: positive in the moral aesthetic sense (independently of the accidental structure of the world). Only then the at. time. It must also mean "attribution" as opposed to privation (or containing privation). This interpretation fits well

If φ pos. and ¬(x) N ¬φ(x) Otherwise φ(x) ⊃_N x ≠ x

hence x ≠ x positive or x = x by contrary Ax. by the existence of one Ax.

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

Formalisation of Gödel's manuscript of 1970 (adapted property inclusion and generalized axiom Ax1Gen: **Gödel3**)

— proof in KB —

$\varphi \supset \Box\Diamond\varphi$

Note: This is a new variant

```
theory GoedelVariantHOML3 imports HOMLinHOL ModalFilter
begin
 consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
 axiomatization where Ax1: "⌊P φ ∧ P ψ ⊃ P (φ • ψ)⌋"
 axiomatization where Ax2a: "⌊P φ ∨ᵉ P ¬φ⌋"
 definition God ("G") where "G x ≡ ∀φ. P φ ⊃ φ x"
(**) abbreviation PropertyInclusion ("_⊃ₙ_") where "φ ⊃ₙ ψ ≡ □(φ ≠ (λx. ⊥) ∧ ∀ᴱy. φ y ⊃ ψ y))"
 definition Essence ("_Ess._") where "φ Ess. x ≡ ∀ψ. ψ x ⊃ (φ ⊃ₙ ψ)"
 axiomatization where Ax2b: "⌊P φ ⊃ □ P φ⌋"
 lemma Ax2b': "⌊¬P φ ⊃ □(¬P φ)⌋" using Ax2a Ax2b by blast
 theorem Th1: "⌊G x ⊃ G Ess. x⌋" using Ax2a Ax2b Essence_def God_def by (smt (verit))
 definition NecExist ("E") where "E x ≡ ∀φ. φ Ess. x ⊃ □(∃ᴱx. φ x)"
 axiomatization where Ax3: "⌊P E⌋"
 theorem Th2: "⌊G x ⊃ □(∃ᴱy. G y)⌋" using Ax3 Th1 God_def NecExist_def by smt
 theorem Th3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" sledgehammer(Th2 Rsymm) —‹Proof found›
   proof -
     have 1: "⌊(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using Th2 by blast
     have 2: "⌊◇(∃ᴱx. G x) ⊃ ◇□(∃ᴱy. G y)⌋" using 1 by blast
     have 3: "⌊◇(∃ᴱx. G x) ⊃ □(∃ᴱy. G y)⌋" using 2 Rsymm by blast
     thus ?thesis by blast
   qed
 axiomatization where Ax4: "⌊P φ ∧ (φ ⊃ₙ ψ) ⊃ P ψ⌋"
 lemma True nitpick[satisfy,card=1,eval="⌊P (λx. ⊥)⌋"] oops —‹Two models found of cardinality one›
 abbreviation "PosProps Φ ≡ ∀φ. Φ φ ⊃ P φ"
 abbreviation "ConjOfPropsFrom φ Φ ≡ □(∀ᴱz. φ z ↔ (∀ψ. Φ ψ ⊃ ψ z))"
 axiomatization where Ax1Gen: "⌊(PosProps Φ ∧ ConjOfPropsFrom φ Φ) ⊃ P φ⌋"
 lemma L:      "⌊P G⌋" using Ax1Gen God_def by (smt (verit))
 theorem Th4: "⌊◇(∃ᴱx. G x)⌋" sledgehammer[timeout=200](Ax2a L Ax1Gen) sorry —‹Proof found›
 theorem Th5: "⌊□(∃ᴱx. G x)⌋" using Th3 Th4 by blast
 lemma MC:    "⌊φ ⊃ □φ⌋" sledgehammer(Ax2a Ax2b Th5 God_def Rsymm) —‹Proof found›
   proof - {fix w fix Q
     have 1: "∀x.(G x w ⟶ (∀Z. Z x ⊃ □(∀ᴱz. G z ⊃ Z z)) w)" using Ax2a Ax2b God_def by smt
     have 2: "(∃x. G x w)⟶((Q ⊃ □(∀ᴱz. G z ⊃ Q)) w)" using 1 by force
     have 3: "(Q ⊃ □Q) w" using 2 Th5 Rsymm by blast}
     thus ?thesis by auto
   qed
 lemma PosProps:    "⌊P (λx.⊤) ∧ P(λx. x = x)⌋"    using Ax2a Ax4 L Th4 by smt
 lemma NegProps:    "⌊¬P(λx.⊥) ∧ ¬P(λx. x ≠ x)⌋" using Ax2a Ax4 L Th4 by smt
 lemma UniqueEss1: "⌊φ Ess. x ∧ ψ Ess. x ⊃ □(∀ᴱy. φ y ↔ ψ y)⌋" oops —‹Unclear, open question›
 lemma UniqueEss2: "⌊φ Ess. x ∧ ψ Ess. x ⊃ □(φ ≡ ψ)⌋" oops —‹Unclear, open question›
 lemma UniqueEss3: "⌊φ Ess. x ⊃ □(∀ᴱy. φ y ⊃ y ≡ x)⌋" using Essence_def MC by auto
 lemma Monotheism: "⌊G x ∧ G y ⊃ x ≡ y⌋" using Ax2a God_def by smt
 lemma Filter:      "⌊Filter P⌋"    using Ax1 Ax4 MC NegProps PosProps Rsymm by smt
 lemma UltraFilter: "⌊UFilter P⌋" using Ax2a Filter by smt
 lemma True nitpick[satisfy,card=1,eval="⌊P (λx. ⊤)⌋"] oops —‹One model found of cardinality one›
end
```

Fig. 8 After an appropriate modification of the notion of necessary property inclusion in Gödel's 1970 ontological proof, the inconsistency revealed in Fig. 6 is avoided, and the argument can be successfully verified in modal logic S5 (indeed, as shown, only symmetry of the accessibility relation is actually needed)

Formalisation of Gödel's manuscript of 1970 (adapted property inclusion and generalized axiom Ax1Gen: **Gödel3**)

— proof in KB —

$\varphi \supset \Box\Diamond\varphi$

Gödel 3 — Results

“Teufel = … (leere Menge)” (Max III, S. 47)

“Bemerkung: Die leere Menge (der leere Raum) ist ein Amphibium zwischen Nichts und Etwas.” (Max III, S. 85)

‘φ necessarily implies ψ’ iff $\Box\forall^E x\,.\,\varphi \neq \lambda x\,.\,\bot \wedge \varphi x \supset \psi x$

The empty set now no longer implies any other properties!

Ax1Gen: The conjunction **of arbitrarily many** positive properties is positive.
Ax2a: Either a property or its complement is positive.
G(od-like): x is God-like iff it possesses all positive properties.
Ess(ence): φ is an essence of x iff all properties of x are necessarily implied by φ.
Ax2b: Positive properties are necessarily positive.
Theorem 1: Being God-like is the essence of every God-like entity.
E(xistence): x has existence iff all essences of x are necessarily exemplified.
Ax3: E(xistence) is a positive property.
Theorem 2: If x is God-like, then God-likeness is necessarily exemplified.
Theorem 3: The possible existence of a God-like entity implies the necessary existence of a God-like entity.
Ax4: Properties that are necessarily implied by a positive property are positive.
Lemma L: God-likeness is a positive property.
Theorem 4: A God-like entity possibly exists.
Theorem 5: A God-like entity necessarily exists.

- Consistent
- Valid already in modal logic **KB**
- **Modal collapse** follows
- **Monotheism** follows
- **P** is an **ultrafilter**
- Self-identity: positive property
- Self-difference: non-positive prop.

Gödel 3 — Results

“Teufel = … (leere Menge)” (Max III, S. 47)

“Bemerkung: Die leere Menge (der leere Raum) ist ein Amphibium zwischen Nichts und Etwas.” (Max III, S. 85)

‘φ necessarily implies ψ’ iff $\Box\forall^E x\,.\,\varphi \neq \lambda x\,.\,\bot \wedge \varphi x \supset \psi x$

The empty set now no longer implies any other properties!

Simplified Variant of Gödel's Ontological Argument

explored through systematic simplification experiments with a modern higher-order theorem proving technology

Simplified Axioms

A1' (C1): Self-difference (empty property) is not a positive property.

A2' (C2): A property is positive if it is entailed by a positive property.

A3: Being Godlike is a positive property.*

(Modal logic K: Necessitation rule N, distribution axiom K)

Single Definition/Shorthand:

D1: An entity x is Godlike if it possesses all positive properties.

Further Findings

- Consistent (model finder Nitpick: one world, one Godlike being)
- Modal collapse not implied (countermodel by Nitpick: two possible worlds i1 and i2 and with one Godlike entity e; i1 and i2 are reachable from i2, but only i1 can be reached from i1; there is a non-positive property ϕ which holds for e in i2 but not in i1)
- But also the attempt to prove that the existence of a Godlike being is possible fails (countermodel by Nitpick: single world with no other world reachable)
- Way out: we simply postulate the additional modal axiom T (what necessarily holds, that holds; resp. what holds is possible)

Simplified Argument (mainly non-modal, except for last step; note also that D1 is not used)

L1: The existence of a non-exemplified positive property implies that self-difference (resp. the empty property) is positive.
(from A2', since such a non-exemplified positive property would entail self-difference (p entails q: for all x, p(x) implies q(x))

L2: A non-exemplified positive property does not exist. (from A1' and the contrapositive of L1)

T1': Positive properties are exemplified. (reformulation of L2)

T4: There exists a Godlike being. (from A3, T1')

T3: Necessarily, there exists a Godlike being. (from T4, by necessitation)

* Alternatively, we may postulate A3': The conjunction of any collection of positive properties is positive; then we prove A3 from A3' and D1.

Trivialising simplifications — how I also produced them — are being eliminated this way.

Symbolic AI and Gödel's Ontological Argument. Zygon 57(4):953–962. Doi: 10.1111/zygo.12830

A Simplified Variant of Gödel’s Ontological Argument. In: A. Vestrucci (ed.), Beyond Babel: Religions and Linguistic Pluralism. Doi: 10.1007/978-3-031-42127-3_19

Gödel 2

Summary

requires the generalized axiom **Ax1Gen**

- modified: **essence**
- no existential import
- consistent
- possible and necessary existence of G
- verified already in logic **KB**
- derived results:
  - **modal collapse**
  - **monotheism**
  - **P is an ultrafilter**

Gödel 3

Summary

requires the generalized axiom **Ax1Gen**

- modified: property **inclusion**
- no existential import
- consistent
- possible and necessary existence of G
- verified already in logic **KB**
- derived results:
  - **modal collapse**
  - **monotheism**
  - **P is an ultrafilter**

Changing the quantifiers for entities from
‘varying-domain’ to ‘constant-domain’? —> results reconfirmed

And the Devil?

“Nämlich, um ein Ding {a} zu erkennen, muss ich haben: 1.) gewisse fundamentale Zuordnungen gewisser empirischer Dinge (vielleicht beginnen mit Gott = Allmenge*; Teufel = ein unwiderlegbar falsches System* (oder leere Menge); …” (Max III, S. 47)

Bem<erkung> (*Theol<ogie>*): Die vollkommen klare Erkenntnis des {vollkommen} Bösen, dass er seine Absicht nie erreichen kann, bedeutet seinen Tod. (Max IX, S. 36)

And the **Devil**? — Variant 1

(Credits to Chad Brown)

```
1  theory ThereIsNoEvil1 imports GoedelVariantHOML2
2  begin
3   definition Evil ("Evil") where "Evil x ≡ ∀φ. ¬ P φ ⊃ φ x"
4   theorem NecNoEvil: "⌊□(¬(∃ᴱx.  Evil x))⌋" sledgehammer(Ax2a Ax4 Evil_def)  —‹Proof found›
5     proof -
6       have   "⌊¬P(λy.⊥)⌋"  using Ax2a Ax4 by blast
7       hence  "⌊(∀ᴱx.  Evil x ⊃ (λy.⊥) x)⌋" using Evil_def by auto
8       hence  "⌊(∀ᴱx.  Evil x ⊃ ⊥)⌋" by auto
9       hence  "⌊(∃ᴱx.  Evil x) ⊃ ⊥⌋" by auto
10      hence  "⌊¬(∃ᴱx. Evil x)⌋" by blast
11      thus ?thesis by blast
12    qed
13 end
```

Gödel 2

Fig. 10 Importing Gödel's modified axioms from Fig. 7 we can prove that necessarily there exists no entity that possesses all non-positive (=negative) properties

x is a Devil iff
x possesses all
non-positive
properties.

+

Axioms as before

↓

Necessarily, no Devil
exists.

```
1  theory ThereIsNoEvil2 imports GoedelVariantHOML3
2  begin
3   definition Evil ("Evil") where "Evil x ≡ ∀φ. ¬ P φ ⊃ φ x"
4   theorem NecNoEvil: "⌊□(¬(∃ᴱx.  Evil x))⌋" sledgehammer(Ax1Gen Ax2a Evil_def) oops  —‹Proof found›
5  end
```

Gödel 3

Fig. 11 Importing Gödel's modified axioms from Fig. 8 we can prove that necessarily there exists no entity that possesses all non-positive (=negative) properties

Also confirmed for
Gödel 3

And the **Devil**? — Variant 2 (Scott)

```
theory EvilDerivable imports HOMLinHOL ModalFilter
begin
 consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
 definition Evil ("Evil") where "Evil x ≡ ∀φ. ¬ P φ ⊃ φ x"
 definition Essence ("_Ess._") where "φ Ess. x ≡  φ x ∧ (∀ψ. ψ x ⊃ □(∀E y.  φ y ⊃ ψ y))"
 definition NecExist ("E") where "E x  ≡ ∀φ. φ Ess. x ⊃ □(∃E x. φ x)"
 axiomatization where A1: "⌊¬P φ ↔ P ~φ⌋"
 axiomatization where A2: "⌊¬P φ ∧ □(∀E y.  φ y ⊃ ψ y) ⊃ ¬P ψ⌋"
 axiomatization where A4: "⌊¬P Evil⌋"
 axiomatization where A3: "⌊¬P φ ⊃ □ (¬P φ)⌋"
 axiomatization where A5: "⌊¬P E⌋"
 lemma True nitpick[satisfy,card i=1,eval="⌊P (λx.⊥)⌋",eval="⌊P (λx. T)⌋"] oops —‹Model found›
 theorem T1: "⌊¬P φ ⊃ ◇(∃E x. φ x)⌋"              using A1 A2 by blast
 theorem T2: "⌊◇(∃E x. Evil x)⌋"                  using A4 T1 by blast
 theorem T3: "⌊Evil x ⊃ Evil Ess. x⌋"             using A1 A3 Essence_def Evil_def  by (smt (verit, best))
 theorem T4: "⌊◇(∃E x. Evil x) ⊃ □(∃E y. Evil y)⌋" using A5 Evil_def NecExist_def Rsymm T3 by smt
 theorem T5: "⌊□(∃E x.  Evil x)⌋"    using T2 T4 by presburger
 lemma MC:          "⌊φ ⊃ □φ⌋"  sledgehammer(A1 A3 T5 Evil_def Rsymm) oops —‹proof found›
 lemma PosProps:    "⌊P (λx.⊥)  ∧  P (λx. x ≠ x)⌋"     using A1 A2 by blast
 lemma NegProps:    "⌊¬P(λx. T) ∧ ¬P(λx. x = x)⌋"  using A1 A2 by blast
 lemma UniqueEss1:  "⌊φ Ess. x ∧ ψ Ess. x ⊃  □(∀E y.  φ y ↔ ψ y)⌋" using Essence_def  by smt
 lemma UniqueEss2:  "⌊φ Ess. x ∧ ψ Ess. x ⊃  □(φ = ψ)⌋" nitpick[card i=2] oops —‹Countermodel found›
 lemma Monoevilism: "⌊Evil x ∧ Evil y ⊃  x ≡ y⌋" using A1 Evil_def by smt
 lemma Filter:      "⌊Filter (λ φ. ¬P φ)⌋"    using A1 Evil_def Rsymm T1 T5  by (smt (verit, best))
 lemma UltraFilter: "⌊UFilter (λ φ. ¬P φ)⌋"  using Filter A1 by blast
end
```

Fig. 20 The necessary existence of an Evil-like entity proved from (controversially) modified assumptions

x is a Devil iff x possesses all non-positive properties

+

Mirroring of all axioms (Scott) for non-positive properties

↓

Necessarily, there exists a Devil

But: The properties of this Devil are absolutely identical to the properties of Gödel's God-like entity!

Gödel's scriptum variant (1970) and mathematical realism

God = (maximally) infinite/unfathomable?

Bem<erkung> (*Phil<osophie>*): Wenn man die Objekte der Mat<hematik> als durch den Geist konstruiert ansieht, bringt man notwendig ein zeitlich<es> Element hinein**.**

(Max IX, S. 45)

"Gott = Allmenge"

(Max III, S. 47)

Bem<erkung> (*Gr<undlagen>*): Ist das universelle ‚Alle' begrifflich früher [d. h. zum Verständnis notwendig] des Speziellen, zum Beispiel**:** alle Zahlen**?** Es scheint nicht, sondern man lernt das Alle sogar an endlichen Beispielen [z. B.**:** alle Spielsteine auf diesem Tisch] und erreicht das universelle <Alle> überhaupt niemals**.**

(Max IX, S. 22)

Gödel 2 — as a starting point …

```
theory GoedelVariantHOML2 imports HOMLinHOL ModalFilter ChurchTTinHOML
begin
 consts PositiveProperty::"(e⇒σ)⇒σ" ("P")
 axiomatization where                                   Ax1: "⌊P φ ∧ P ψ ⊃ P (φ . ψ)⌋"
 axiomatization where                                  Ax2a: "⌊P φ ∨ᵃ P ~φ⌋"
 definition God ("G") where                                  "G x = ∀φ. P φ ⊃ φ x"
 abbreviation PropertyInclusion ("_⊃ₙ_") where               "φ ⊃ₙ ψ ≡ □(∀ᶠy. φ y ⊃ ψ y)"
 definition Essence ("_Ess._") where                         "φ Ess. x ≡ φ x ∧ (∀ψ. ψ x ⊃ (φ ⊃ₙ ψ))"
 axiomatization where                                  Ax2b: "⌊P φ ⊃ □ P φ⌋"
 lemma          Ax2b': "⌊¬P φ ⊃ □(¬P φ)⌋"          using Ax2a Ax2b by blast
 theorem         Th1: "⌊G x ⊃ G Ess. x⌋"           using Ax2a Ax2b Essence_def God_def by (smt (verit))
 definition NecExist ("E") where                             "E x  = ∀φ. φ Ess. x ⊃ □(∃ᴸx. φ x)"
 axiomatization where                                   Ax3: "⌊P E⌋"
 theorem         Th2: "⌊G x ⊃ □(∃ᴸy. G y)⌋"         using Ax3 Th1 God_def NecExist_def by smt
 theorem         Th3: "⌊◇(∃ᶠx. G x) ⊃ □(∃ᶠy. G y)⌋"  by (metis Th2 Rsymm)
 axiomatization where                                   Ax4: "⌊P φ ∧ (φ ⊃ₙ ψ) ⊃ P ψ⌋"
 lemma True nitpick[satisfy,card=1,eval="⌊P (λx.⊥)⌋"] oops —‹One model found of cardinality one›
 abbreviation    "CollOfPosProps Φ = ∀φ. Φ φ ⊃ P φ"
 abbreviation    "ConjOfPropsFrom φ Φ ≡ □(∀ᶠz. φ z ↔ (∀ψ. Φ ψ ⊃ ψ z))"
 axiomatization where                                 Ax1Gen: "⌊(CollOfPosProps Φ ∧ ConjOfPropsFrom φ Φ) ⊃ P φ⌋"
 lemma            L: "⌊P G⌋"                       using Ax1Gen God_def by (smt (verit))
 theorem         Th4: "⌊◇(∃ᴸx. G x)⌋"              using Ax2a Ax4 L by blast
 theorem         Th5: "⌊□(∃ᶠx. G x)⌋"              using Th3 Th4 by blast
 lemma           MC: "⌊φ ⊃ □φ⌋"  —‹Proof found with sledgehammer›
   proof - {fix w fix Q
    have 1: "∀x.(G x w     › (∀Z. Z x ⊃ □(∀ᴸz. G z ⊃ Z z)) w)" using Ax2a Ax2b God_def by smt
    have 2: "(∃x. G x w)⟶((Q ⊃ □(∀ᶠz. G z ⊃ Q)) w)" using 1 by force
    have 3: "(Q ⊃ □Q) w" using 2 Th5 Rsymm by blast}
    thus ?thesis by auto qed
 lemma       PosProps: "⌊P (λx.⊤) ∧ P (λx. x = x)⌋"  using Ax2a Ax4 by blast
 lemma       NegProps: "⌊¬P(λx.⊥) ∧ ¬P(λx. x ≠ x)⌋"  using Ax2a Ax4 by blast
 lemma     UniqueEss1: "⌊φ Ess. x ∧ ψ Ess. x ⊃ □(∀ᴸy. φ y ↔ ψ y)⌋"      using Essence_def by smt
 lemma     UniqueEss2: "⌊φ Ess. x ∧ ψ Ess. x ⊃ □(φ ≡ ψ)⌋"  nitpick[card i=1] oops —‹Countermodel found›
 lemma     Monotheism: "⌊G x ∧ G y ⊃ x ≡ y⌋"          using Ax2a God_def by smt
 lemma         Filter: "⌊Filter P⌋"                   using Ax1 Ax4 MC NegProps PosProps Rsymm by smt
 lemma     UltraFilter: "⌊UFilter P⌋"                 using Ax2a Filter by smt
 lemma True nitpick[satisfy,user_axioms,card=1,eval="⌊P (λx.⊥)⌋"] oops —‹One model found of cardinality one›
```

…

Gödel 2 — plus mathematical objects

Assumption: The objects of mathematics ‘exist’ (e.g. natural numbers n, Peano axioms).

Predicates: $\lambda x\,.\,x = n$ **and** $\lambda x\,.\,x \neq n$

Implications for the cardinality of the positive properties P:

- **P** is infinit**e**
- **P** is **uncountably infinite**

Claim (verification is further work):

if we again assume such properties to exist, then the generalized axiom Ax1Gen forces the cardinality of P into the maximally infinite

...

```
38  —‹Assuming 2 different entities leads to 2 different positive properties, assuming 3 leads to 4, etc.›
39  lemma H: "⌊P (λe. ⊤) ∧ (P (λe. e=a) ∨ P (λe. e≠a))⌋" using PosProps Ax2a by auto
40  lemma PP2: assumes "⌊∃(a::e) b. a ≠ b⌋" shows "⌊∃p q. p≠q ∧ P p ∧ P q⌋" by (smt (verit, del_insts) H assms)
41  lemma PP4: assumes "⌊∃(a::e) b c. a ≠ b ∧ a ≠ c ∧ b ≠ c⌋"
42    shows "⌊∃p q r s. p≠q ∧ p≠r ∧ p≠s ∧ q≠r ∧ q≠s ∧ r≠s ∧ P p ∧ P q ∧ P r ∧ P s⌋" by (smt (verit, del_insts) H assms)
43  —‹Assuming infinitely many different entities leads to at least 2, 4, etc., many different positive properties›
44  lemma PP2': assumes "⌊Inf0⌋" shows "⌊∃p q. ((p≠q) ∧ P p ∧ P q)⌋" by (metis (lifting) Ax2a Inf0_def PosProps assms)
45  lemma PP4': assumes "⌊Inf0⌋" shows "⌊∃p q r s. p≠q∧p≠r∧p≠s∧q≠r∧q≠s∧r≠s∧P p∧P q∧P r∧P s⌋"
46   proof - have "⌊Inf0⌋ ⟹ ⌊∃p::e. ∃q::e. ∃r::e. p≠q ∧ p≠r ∧ q≠r⌋" unfolding Inf0_def apply simp by metis
47     thus ?thesis using PP4 assms by blast qed
48  —‹Theorem MetaInf: infinitely many different entities lead to infinitely many positive properties: proof at meta-level.›
49   —‹Various definitions and automatically proved lemmata are not shown here.›                    (*{{{ *) [9 lines]
59  lemma MetaInf: "⌊Inf0⌋ ⟹ infinite (UNIV::e set)"                                              (*{{{ *) [6 lines]
66  lemma InfPosSet_a: "infinite (UNIV::e set) ⟹ infinite (𝒫 w)"                                   (*{{{ *) [5 lines]
72  lemma InfPosSet_b: "⌊Inf0⌋ ⟹ infinite (𝒫 w)" using InfPosSet_a MetaInf by blast              (*{{{ *) [6 lines]
79  theorem InfPosSet: assumes "⌊Inf0⌋" shows "⌊¬(Finite P)⌋" by (metis Finite_def InfPosSet_b Nat_finite assms)
80  —‹CantorArgument: Assuming infinitely many different entities leads to uncountably many different positive properties.›
81  definition "S (f::nat⇒e) ≡ (λx::e. if x ∈ range f then f (Suc (inv f x)) else x)"
82  definition "Enc (f::nat⇒e) s = (λ(B::e⇒bool) (x::e) (u::i). x = f 0 ∨ (∃y. s y = x ∧ B y))"
83  definition "D F (f::nat⇒e) s (w::i) ≡ (λx::e. if F x (f 0) w then ¬ F x (s x) w else F x (s x) w)"
84  lemma Inf_inj: "⌊Inf0⌋ ⟹ ∃f::nat⇒e. inj f" using MetaInf infinite_countable_subset by blast
85  lemma Enc0: "Enc f s B (f 0) w" using Enc_def by simp
86  lemma EncS: "inj f ⟶ (Enc f (S f) B (S f y) w ⟷ B y)"
87   by (smt Enc_def S_def inj_Suc inv_f_f inv_into_injective nat.distinct rangeI)
88  lemma L1: "inj (f::nat⇒e) ⟹ (¬(∃F. ∀p. P p ⊃ (∃e::e. F e = p))) w" by (smt Enc0 EncS Ax2a D_def)
89  theorem CantorArgument: assumes "⌊Inf0⌋" shows "⌊¬(∃F. ∀p. P p ⊃ (∃e::e. F e = p))⌋" using assms Inf_inj L1 by metis
90  end
```

On the maximality of positive properties and modal collapse in variants of Gödel's ontological proof of God

Cordelia Mühlenbeck, Christoph Benzmüller · *Logic and Logical Philosophy*, 1–21

DOI ↗ URL ↗ BibTeX

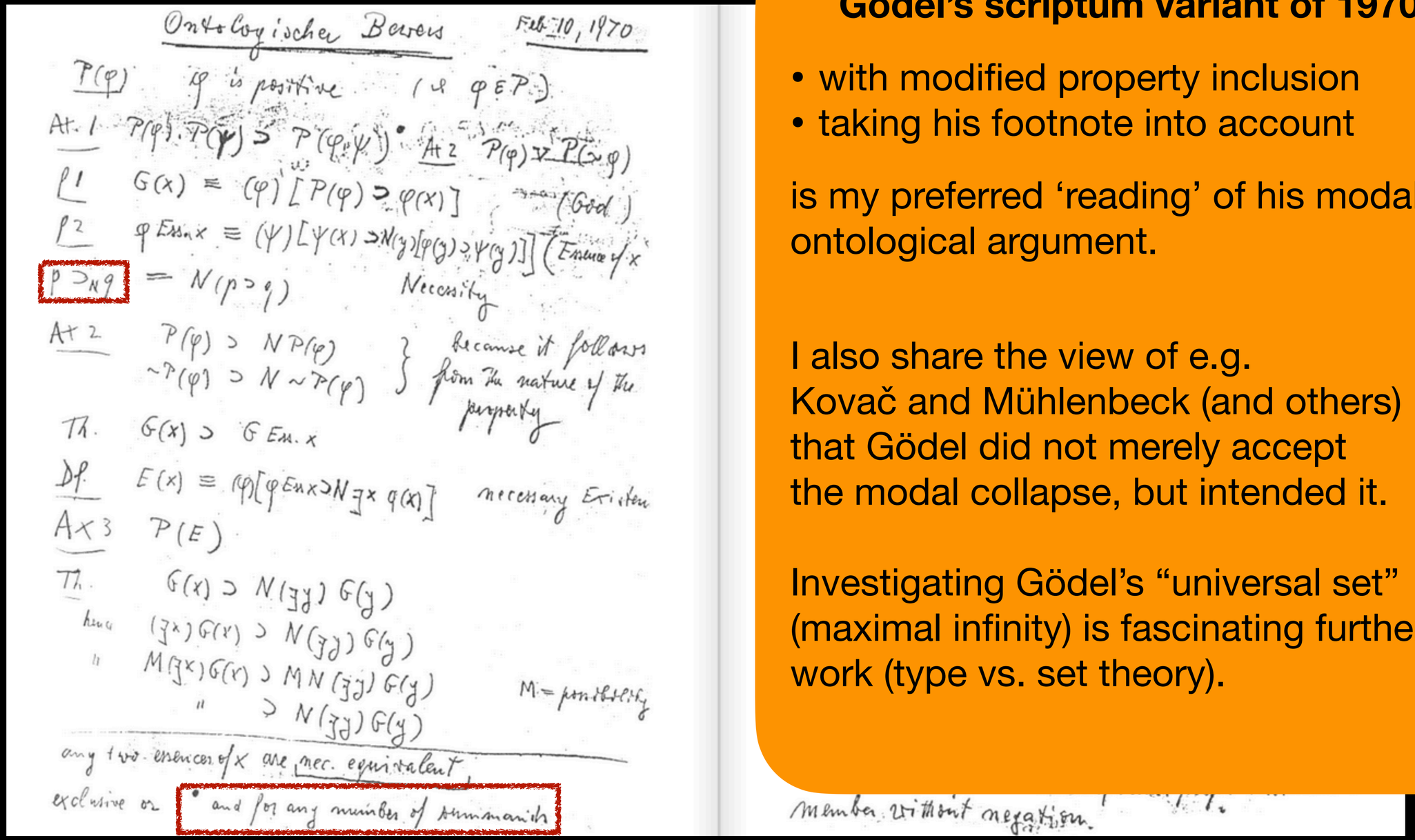
On Absolute Infinity and Modal Collapse in Gödel's Ontological Proof

Cordelia Mühlenbeck, Christoph Benzmüller · *Logic, Knowledge, and Tradition: Essays in Honor of Srećko Kovač* · In print

BibTeX

```
22  —‹Formalized notions: equipollence, cardinality, different notion of infinity›
23  abbreviation Ex1B (binder "∃₁" [8]9) where "∃₁x. Φ(x) ≡ (∃x. Φ(x) ∧ (∀z. Φ(z) ⊃ (z = x)))"
24  definition "Equipollent ≡ (λp::'a⇒σ. λq::'b⇒σ. (∃s. (∀x. p x ⊃ q (s x)) ∧ (∀y. q y ⊃ (∃₁x. p x ∧ (y = s x)))))"
25  definition "Cardinality ≡ (λp::'b⇒σ. Equipollent p)"
26  definition "CardinalNumbers ≡ (λu::(('b⇒σ)⇒σ) . (∃p::('b⇒σ). (u = (Equipollent p))))"
27  definition "Inf0 ≡ ∃r::e⇒e⇒σ. ∀x y z. ((∃w. r x w) ∧ ¬ (r x x) ∧ (r x y ⊃ (r y z ⊃ r x z)))"
28  definition "Inf1 ≡ ∃n::e⇒σ. ∃m::e⇒σ. (∀x. m x ⊃ n x) ∧ (∃u. n u ∧ ¬(m u)) ∧ (Equipollent n m)"
29  definition "Inf2 ≡ ∃n::e⇒σ.∃u.∃s. n u ∧ (∀x. n x ⊃ n (s x)) ∧ (∀x. n x ⊃ (s x)≠u) ∧ (∀x y. (n x ∧ n y ∧ s x=s y) ⊃ x=y)"
30  —‹Definitions/abbreviations for modalised natural numbers (modeled as analysable objects naturally contained in HOL)›
31  named_theorems Nat
32  definition Zero ("0") where [Nat]:      "0 ≡ ((=) (λx. ⊥))"
33  definition Succ ("S") where [Nat]:      "S ≡ (λn. λp. (∃x. p x ∧ (n (λt. (t ≠ x) ∧ p t))))"
34  definition Naturals ("N") where [Nat]:  "N ≡ (λn. (∀p. (p 0 ∧ (∀x. p x ⊃ p (S x))) ⊃ p n))"
35  definition Finite ("Finite") where [Nat]: "Finite ≡ (λp. (∃n. N n ∧ n p))"
```

Personal conclusion



Ontologischer Beweis Feb 10, 1970

P(φ) φ is positive (= φ ∈ P)

Ax. 1 $P(\varphi) \cdot P(\psi) \supset P(\varphi \cdot \psi)$ • Ax 2 $P(\varphi) \vee P(\sim\varphi)$

D1 $G(x) \equiv (\varphi)[P(\varphi) \supset \varphi(x)]$ (God)

D2 $\varphi \text{ Ess. } x \equiv (\psi)[\psi(x) \supset N(y)[\varphi(y) \supset \psi(y)]]$ (Essence of x)

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

Ax 2 $P(\varphi) \supset NP(\varphi)$ } because it follows from the nature of the property

$\sim P(\varphi) \supset N \sim P(\varphi)$ }

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

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

Ax 3 $P(E)$

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

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

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

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

any two essences of x are nec. equivalent

exclusive or • and for any number of summands

member without negation.

Gödel's scriptum variant of 1970

- with modified property inclusion
- taking his footnote into account

is my preferred 'reading' of his modal ontological argument.

I also share the view of e.g. Kovač and Mühlenbeck (and others) that Gödel did not merely accept the modal collapse, but intended it.

Investigating Gödel's "universal set" (maximal infinity) is fascinating further work (type vs. set theory).

References

The ontological argument — original & key sources

- Gödel, K. (2004/1970). **Appendix A: Notes in Kurt Gödel's Hand**. In: J.H. Sobel, Logic and Theism, pp. 144–145. Cambridge University Press.
- Scott, D.S. (2004/1970). **Appendix B: Notes in Dana Scott's Hand**. In: J.H. Sobel, Logic and Theism, pp. 145–146. Cambridge University Press. Doi: 10.1017/CBO9780511497988
- Sobel, J.H. (2004). **Logic and Theism: Arguments for and Against Beliefs in God**. Cambridge University Press. Doi: 10.1017/CBO9780511497988
- Adams, R.M. (1995). **Introductory Note to *1970**. In: Kurt Gödel: Collected Works, vol. III, pp. 388–402. Oxford University Press. Doi: 10.1093/oso/9780195072556.003.0017
- Kanckos, A. & Lethen, T. (2021). **The Development of Gödel's Ontological Proof**. Review of Symbolic Logic 14(4):1011–1029. Doi: 10.1017/S1755020319000479
- Wang, H. (1997). **A Logical Journey: From Gödel to Philosophy**. MIT Press. Doi: 10.7551/mitpress/4321.001.0001
- Gödel, K. (2019 ff.). **Philosophische Notizbücher / Philosophical Notebooks** (ed. E.-M. Engelen, transl. M. Carl). De Gruyter, Berlin/Boston. Doi: 10.1515/9783110583748 (vol. 1); here Max Phil 0, II–IX.

References

Emendations, variants & criticism

- Anderson, C.A. (1990). **Some Emendations of Gödel's Ontological Proof**. Faith and Philosophy 7(3):291–303. Doi: 10.5840/faithphil19907325
- Anderson, C.A. & Gettings, M. (1996). **Gödel's Ontological Proof Revisited**. In: Gödel '96, Lecture Notes in Logic 6, pp. 167–172. Cambridge University Press. Doi: 10.1017/9781316716939.011
- Hájek, P. (1996). **Magari and Others on Gödel's Ontological Proof**. In: Logic and Algebra, pp. 125–135. Doi: 10.1201/9780203748671
- Hájek, P. (2002). **A New Small Emendation of Gödel's Ontological Proof**. Studia Logica 71(2):149–164. Doi: 10.1023/A:1016583920890
- Fitting, M. (2002). **Types, Tableaus, and Gödel's God**. Springer. Doi: 10.1007/978-94-010-0411-4
- Kovač, S. (2012). **Modal Collapse in Gödel's Ontological Proof**. In: Ontological Proofs Today, pp. 323–343. Ontos. Doi: 10.1515/9783110325881.323
- Hazen, A.P. (1998). **On Gödel's Ontological Proof**. Australasian Journal of Philosophy 76(3):361–377. Doi: 10.1080/00048409812348501
- Bjørdal, F.A. (2018). **All Properties Are Divine or God Exists**. Logic and Logical Philosophy 27(3). Doi: 10.12775/LLP.2017.027

References

Computer-supported analysis of Gödel’s argument

- Benzmüller, C. & Scott, D.S. (2025). **Notes on Gödel's and Scott's Variants of the Ontological Argument**. Monatshefte für Mathematik. Doi: 10.1007/s00605-025-02078-x
- Benzmüller, C. & Woltzenlogel Paleo, B. (2014). **Automating Gödel's Ontological Proof of God's Existence with Higher-order Automated Theorem Provers**. ECAI 2014. Doi: 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)
- Benzmüller, C., Weber, L. & Woltzenlogel Paleo, B. (2017). **Computer-Assisted Analysis of the Anderson–Hájek Controversy**. Logica Universalis 11(1):139–151. Doi: 10.1007/s11787-017-0160-9
- Fuenmayor, D. & Benzmüller, C. (2017). **Automating Emendations of the Ontological Argument in Intensional Higher-Order Modal Logic**. KI 2017. Doi: 10.1007/978-3-319-67190-1_9
- Benzmüller, C. & Fuenmayor, D. (2020). **Computer-Supported Analysis of Positive Properties, Ultrafilters and Modal Collapse in Variants of Gödel's Ontological Argument**. Bulletin of the Section of Logic 49(2):127–148. Doi: 10.18778/0138-0680.2020.08
- Benzmüller, C. (2022). **Symbolic AI and Gödel's Ontological Argument**. Zygon 57(4):953–962. Doi: 10.1111/zygo.12830
- __ (2023). **A Simplified Variant of Gödel’s Ontological Argument**. In: A. Vestrucci (ed.), Beyond Babel: Religions and Linguistic Pluralism. Springer. Doi: 10.1007/978-3-031-42127-3_19
- Vestrucci, A. & Benzmüller, C. (2024). **Reflections on the Ontological Proof of Kurt Gödel**. In: Divined Explanations, S. 255–285. Brill. Doi: 10.1163/9789004701908_013
- Mühlenbeck, C. & Benzmüller, C. (2026). **On the maximality of positive properties and modal collapse in variants of Gödel’s ontological proof of God**. Logic and Logical Philosophy. Doi: 10.12775/LLP.2026.007

References

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

Methodological foundations (higher-order logic, modal logic, tools)

- Church, A. (1940). **A Formulation of the Simple Theory of Types**. J. Symb. Logic 5:56–68. Doi: 10.2307/2266170
- Henkin, L. (1950). **Completeness in the Theory of Types**. J. Symb. Logic 15(2):81–91. Doi: 10.2307/2266967
- Benzmüller, C., Brown, C. & Kohlhase, M. (2004). **Higher-Order Semantics and Extensionality**. J. Symb. Logic 69(4):1027–1088. Doi: 10.2178/jsl/1102022211
- Benzmüller, C. & Andrews, P. (2024). **Church's Type Theory**. Stanford Encyclopedia of Philosophy. (https://plato.stanford.edu/entries/type-theory-church/)
- Benzmüller, C. & Paulson, L. (2013). **Quantified Multimodal Logics in Simple Type Theory**. Logica Universalis 7(1):7–20. Doi: 10.1007/s11787-012-0052-y
- Scott, D.S. (1967). **Existence and Description in Formal Logic**. In: Bertrand Russell: Philosopher of the Century, pp. 181–200. (Reprinted in: Philosophical Applications of Free Logic, OUP 1991)
- Nipkow, T., Paulson, L.C. & Wenzel, M. (2002). **Isabelle/HOL — A Proof Assistant for Higher-Order Logic**. Springer, LNCS 2283. Doi: 10.1007/3-540-45949-9
- Blanchette, J.C. & Nipkow, T. (2010). **Nitpick: A Counterexample Generator for Higher-Order Logic**. ITP 2010. Doi: 10.1007/978-3-642-14052-5_11
- Steen, A. & Benzmüller, C. (2021). **Extensional Higher-Order Paramodulation in Leo-III**. J. Autom. Reasoning 65(6):775–807. Doi: 10.1007/s10817-021-09588-x