Higher-Order Modal Logics: Automation and Applications

Christoph Benzmüller[1]* and Bruno Woltzenlogel Paleo[2]

[1] Freie Universität Berlin, Germany
c.benzmueller@fu-berlin.de
[2] Vienna University of Technology, Austria
bruno@logic.at

Abstract. *These are the lecture notes of a tutorial on higher-order modal logics held at the 11th Reasoning Web Summer School.* After defining the syntax and (possible worlds) semantics of some higher-order modal logics, we show that they can be embedded into classical higher-order logic by systematically lifting the types of propositions, making them depend on a new atomic type for possible worlds. This approach allows several well-established automated and interactive reasoning tools for classical higher-order logic to be applied also to modal higher-order logic problems. Moreover, also meta reasoning about the embedded modal logics becomes possible. Finally, we illustrate how our approach can be useful for reasoning with web logics and expressive ontologies, and we also sketch a possible solution for handling inconsistent data.

1 Introduction and Overview

Expressivity matters. Often problems can be elegantly encoded and solved in expressive higher-order logics, while their encoding and/or solution in (theoretically or practically) less expressive logics is significantly more involved or even condemned to fail. A prominent example that well illustrates this issue for the transition from first-order to higher-order logic is Boolos' *curious inference* [29] (which has been formalized with modern higher-order proof assistants [15]). In higher-order logic there is a short, one page proof, whereas the corresponding first-order proof is intractably long.

Another, more practical example from mathematics is Cantor's theorem (the set of all subsets of A, that is, the power set of A, has a strictly greater cardinality than A itself). In classical higher-order logic Cantor's theorem (surjective version) can be encoded as $\neg\exists F\forall G\exists X.FX = G$. Higher-order theorem provers can solve this problem very efficiently, and their solution includes the detection and application of the diagonalisation argument [8]. In fact, this theorem is today often used as a very first test example for new higher-order theorem provers. Other illustrating examples include McCarthy's checkerboard problem or the fixed point theorem [9].

Modal logics [26] extend usual formal logic languages by adding modal operators ($\Box$ and $\Diamond$) and are characterized by the *necessitation rule*, according to which $\Box A$ is a

* This work has been supported by the German Research Foundation DFG under grants BE2501/9-1,2 and BE2501/11-1.

theorem if A is a theorem, even though $A \supset \Box A$ is not necessarily a theorem. Various notions, such as *necessity and possibility*, *obligation and permission*, *knowledge and belief*, and *temporal globality and eventuality*, which are ubiquitous in various application domains, have been formalized with the help of modal operators.

In Philosophy, Gödel's modern version of the ontological argument [43, 66] is an interesting example that uses modal operators to express metaphysical necessity and possibility as fundamental notions. In knowledge representation, higher-order logic's expressivity is well-suited to automate meta-logical reasoning about distinct formalisms, such as description logics and modal logics, establishing and verifying correspondence results between them; and, furthermore, some ontologies, such as SUMO [58], could benefit from a reformalisation using modal operators.

Despite the importance of modal logics, general automated reasoning support for them is still not as well-developed as for classical logics. Deduction tools for modal logics are often limited to propositional, quantifier-free fragments or tailored to particular modal logics and their applications; first-order automated deduction techniques based on tableaux, sequent calculi and connection calculi have only recently been generalized and implemented in a few new provers able to directly cope with modalities [55, 17].

Another approach is the embedding of first-order and even higher-order modal logics (HOML) into classical higher-order logics (HOL) [21, 20], for which a range of robust and increasingly effective automated theorem provers has been recently developed [12, 30, 54, 50, 48, 28].

The embedding approach is flexible, because various modal logics (even with multiple modalities or varying/cumulative domain quantifiers) can be easily supported by stating their characteristic axioms. Moreover, the approach is relatively simple to implement, because it does not require any modification in the source code of the higher-order prover. The prover can be used as is, and only the input files provided to the prover must be specially encoded (using lifted versions of connectives and logical constants instead of the usual ones). Furthermore, the efficacy and efficiency of the embedding approach has been confirmed in philosophical benchmarks such as Gödel's ontological argument and some of its variants [56, 13, 19, 24]. These qualities make embedding a convenient approach for *automated* and *interactive* reasoning with propositional and quantified modal logics and possibly many other prominent non-classical logics such as hybrid logics and paraconsistent logics.

In these lecture notes, the syntax and semantics of higher-order logics and higher-order modal logics are introduced and the embedding approach is explained. Then some of the motivating applications described above are explored in greater detail.

2 Higher-Order Modal Logic: Syntax and Semantic

In this section a higher-order modal logic (HOML) is defined by extending a higher-order logic (HOL) with the modal operator $\Box$. An appropriate notion of semantics for HOML is obtained by adapting Henkin semantics for HOL (cf. [45] and [39]). The presentation in this section is borrowed from [19], which adapts [53] and [7].

HOML is a typed logic. More precisely, it is based on Church's simple types. Below only two base types are assumed, but other base types could be easily added.

Definition 1. *The set T of* simple types *is freely generated from the set of basic types* $\{o, \mu\}$ *(*o *stands for Booleans and* μ *for individuals) using the function type constructor* $\rightarrow$*. We may avoid parentheses, and* $\alpha \rightarrow \alpha \rightarrow \alpha$ *then stands for* $(\alpha \rightarrow (\alpha \rightarrow \alpha))$*, that is, function types associate to the right.*

The syntax of the HOML language is given below.

Definition 2. *The* grammar *for HOML is:*

$$\begin{array}{rcl} s, t & ::= & p_\alpha \mid X_\alpha \mid (\lambda X_\alpha . s_\beta)_{\alpha\rightarrow\beta} \mid (s_{\alpha\rightarrow\beta}\, t_\alpha)_\beta \mid (\neg_{o\rightarrow o}\, s_o)_o \mid \\ & & ((\vee_{o\rightarrow o\rightarrow o} s_o)\, t_o)_o \mid (\forall_{(\alpha\rightarrow o)\rightarrow o}(\lambda X_\alpha . s_o))_o \mid (\Box_{o\rightarrow o}\, s_o)_o \end{array}$$

where $\alpha, \beta \in T$*.* p_α *denotes typed constants and* X_α *typed variables (distinct from* p_α*). Complex typed terms are constructed via abstraction and application. The type of each term is given as a subscript. Terms* s_o *of type* o *are called formulas. The logical connectives of choice are* $\neg_{o\rightarrow o}$*,* $\vee_{o\rightarrow o\rightarrow o}$*,* $\forall_{(\alpha\rightarrow o)\rightarrow o}$ *(for* $\alpha \in T$*), and* $\Box_{o\rightarrow o}$*. Type subscripts may be dropped if irrelevant or obvious. Similarly, parentheses may be avoided. Binder notation* $\forall X_\alpha s_o$ *is used as shorthand for* $\forall_{(\alpha\rightarrow o)\rightarrow o}(\lambda X_\alpha . s_o)$*, and infix notation* $s \vee t$ *is employed instead of* $((\vee s)\, t)$*. From the above connectives, other logical connectives, such as* $\top$*,* $\bot$*,* $\wedge$*,* $\supset$*,* $\equiv$*,* $\exists$*, and* $\Diamond$*, can be defined in the usual way.*

Substitution and λ-conversion are defined as usual.

Definition 3. Substitution *of a term* A_α *for a variable* X_α *in a term* B_β *is denoted by* $[A/X]B$*. Since we consider* α*-conversion implicitly, we assume the bound variables of* B *avoid variable capture.*

Definition 4. *Two common relations on terms are given by* β*-reduction and* η*-reduction. A* β*-redex has the form* $(\lambda X . s)t$ *and* β*-reduces to* $[t/X]s$*. An* η*-redex has the form* $(\lambda X . sX)$ *where variable* X *is not free in* s*; it* η*-reduces to* s*. We write* $s =_\beta t$ *to mean* s *can be converted to* t *by a series of* β*-reductions and expansions. Similarly,* $s =_{\beta\eta} t$ *means* s *can be converted to* t *using both* β *and* η*. For each* $s_\alpha \in$ *HOML there is a unique* β-normal form *and a unique* $\beta\eta$-normal form*.*

As a first step towards defining a semantics for HOML, frame structures are introduced. Variables, constants and terms of HOML will subsequently be identified with objects provided in a frame.

Definition 5. *A* frame D *is a collection* $\{D_\alpha\}_{\alpha\in T}$ *of nonempty sets* D_α*, such that* $D_o = \{T, F\}$ *(for truth and falsehood). The* $D_{\alpha\rightarrow\beta}$ *are collections of functions mapping* D_α *into* D_β*.*

Starting from a frame, the notion of a HOML model structure is introduced.

Definition 6. *A* model *for HOML is a quadruple $M = \langle W, R, D, \{I_w\}_{w \in W}\rangle$, where W is a set of worlds (or states), R is an accessibility relation between the worlds in W, D is a frame, and for each $w \in W$, $\{I_w\}_{w \in W}$ is a family of typed interpretation functions mapping constant symbols p_α to appropriate elements of D_α, called the* denotation of p_α in world w *(the logical connectives $\neg$, $\vee$, $\forall$, and $\Box$ are always given the standard denotations, see below). Moreover, it is assumed that the domains $D_{\alpha\to\alpha\to o}$ contain the respective identity relations on objects of type α (to overcome the extensionality issue discussed in [6]).*

Variable assignments are a technical aid for the subsequent definition of an interpretation function $\|.\|^{M,g,w}$ for HOML terms. This interpretation function is parametric over a model M, a variable assignment g and a possible world w.

Definition 7. *A* variable assignment *g maps variables X_α to elements in D_α. $g[d/W]$ denotes the assignment that is identical to g, except for variable W, which is now mapped to d.*

Definition 8. *The* value *$\|s_\alpha\|^{M,g,w}$ of a HOML term s_α on a model $M = \langle W, R, D, \{I_w\}_{w \in W}\rangle$ in a world $w \in W$ under variable assignment g is an element $d \in D_\alpha$ defined in the following way:*

1. $\|p_\alpha\|^{M,g,w} = I_w(p_\alpha)$
2. $\|X_\alpha\|^{M,g,w} = g(X_\alpha)$
3. $\|(s_{\alpha\to\beta}\, t_\alpha)_\beta\|^{M,g,w} = \|s_{\alpha\to\beta}\|^{M,g,w}(\|t_\alpha\|^{M,g,w})$
4. $\|(\lambda X_\alpha . s_\beta)_{\alpha\to\beta}\|^{M,g,w}$ *= the function f from D_α to D_β such that $f(d) = \|s_\beta\|^{M,g[d/X_\alpha],w}$ for all $d \in D_\alpha$*
5. $\|(\neg_{o\to o}\, s_o)_o\|^{M,g,w} = T$ *if and only if* $\|s_o\|^{M,g,w} = F$
6. $\|((\vee_{o\to o\to o}\, s_o)\, t_o)_o\|^{M,g,w} = T$ *if and only if* $\|s_o\|^{M,g,w} = T$ *or* $\|t_o\|^{M,g,w} = T$
7. $\|(\forall_{(\alpha\to o)\to o}(\lambda X_\alpha . s_o))_o\|^{M,g,w} = T$ *if and only if for all $d \in D_\alpha$ we have* $\|s_o\|^{M,g[d/X_\alpha],w} = T$
8. $\|(\Box_{o\to o}\, s_o)_o\|^{M,g,w} = T$ *if and only if for all $v \in W$ with wRv we have* $\|s_o\|^{M,g,v} = T$

Standard semantics does not allow a complete mechanization of HOML. For this reason, Henkin style semantics is introduced here and assumed in the remainder. Henkin semantics allows a complete mechanization of HOML (at least in theory).

Definition 9. *A model $M = \langle W, R, D, \{I_w\}_{w \in W}\rangle$ is called a* standard model *if and only if for all $\alpha, \beta \in T$ we have $D_{\alpha\to\beta} = \{f \mid f : D_\alpha \longrightarrow D_\beta\}$. In a* Henkin model *function spaces are not necessarily full. Instead it is only required that $D_{\alpha\to\beta} \subseteq \{f \mid f : D_\alpha \longrightarrow D_\beta\}$ (for all $\alpha, \beta \in T$) and that the valuation function $\|\cdot\|^{M,g,w}$ from above is total (i.e., every term denotes). Any standard model is obviously also a Henkin model. We consider Henkin models in the remainder.*

Truth in a model, validity in a model M and general validity are defined as usual.

Definition 10. *A formula s_o is* true *in model M for world w under assignment g if and only if $\|s_o\|^{M,g,w} = T$; this is also denoted as $M, g, w \models s_o$. A formula s_o is called* valid *in M if and only if $M, g, w \models s_o$ for all $w \in W$ and all assignments g. Finally, a formula s_o is called* valid*, which we denote by $\models s_o$, if and only if s_o is valid for all M.*

The definitions above introduce higher-order modal logic K. In order to obtain logics KB, KD, S4 and S5, for example, respective conditions on accessibility relation R are postulated: R is a symmetric relation in logic KB, and it is an equivalence relation in logic S5. If these restriction apply, we use the notations $\models^{KB}$ and $\models^{S5}$. In a similar way we may introduce further logics between K and S5, such as D, S4, KD45, etc.

An important issue for quantified modal logics is whether constant domain or varying domain semantics is considered. The definitions above assume constant domains. An adaptation to varying or cumulative domains is straightforward (cf. [37]).

3 Semantic Embedding in Classical Higher-Order Logic

A crucial aspect of modal logics [26] is that the so-called *necessitation rule* allows $\Box A$ to be derived if A is a theorem, but $A \supset \Box A$ is not necessarily a theorem. Naive attempts to define the modal operators $\Box$ and $\Diamond$ may easily be unsound in this respect. To avoid this issue, the *possible world semantics* of modal logics can be explicitly embedded into HOL [21, 20].

The embedding technique described in this section is related to labeling techniques [38]. However, the expressiveness of HOL can be exploited in order to encode the labels within the logical language itself. HOML is embedded into HOL by systematically lifting the types of propositions, making them depend on a new atomic type for possible worlds. This approach allows several well-established automated and interactive reasoning tools for HOL to be applied also to HOML problems. Moreover, also meta reasoning about the embedded modal logics becomes possible [14]. The presentation in this section is adapted from [19] and [17].

3.1 Classical Higher-Order Logic: Syntax and Semantic

HOL is easily obtained from HOML by removing the modal operator $\Box$ from the grammar, and by dropping the set of possible worlds W and the accessibility relation R from the definition of a model. Nevertheless, we explicitly state the most relevant definitions for the particular notion of HOL as employed in this paper. One reason is that we do want to carefully distinguish the HOL and HOML languages in the remainder (we use boldface fonts for HOL and standard fonts for HOML). There is also a subtle, but harmless, difference in the HOL language defined here in comparison to the language in standard presentations: here three base types are employed, whereas usually only two base types are considered. The third base type plays a crucial role in our embedding of HOML in HOL.

Definition 11. *The set $\boldsymbol{T}$ of simple types freely generated from a set of basic types $\{\boldsymbol{o}, \boldsymbol{\mu}, \boldsymbol{\iota}\}$ using the function type constructor $\rightarrow$. $\boldsymbol{o}$ is the type of Booleans, $\boldsymbol{\mu}$ is the type of individuals, and $\boldsymbol{\iota}$ is the type of possible worlds below. As before we may avoid parentheses.*

Definition 12. *The grammar for higher-order logic HOL is:*

$$\begin{aligned} s, t \quad ::= \quad & p_\alpha \mid X_\alpha \mid (\lambda X_\alpha . s_\beta)_{\alpha\rightarrow\beta} \mid (s_{\alpha\rightarrow\beta}\, t_\alpha)_\beta \mid \neg_{o\rightarrow o}\, s_o \mid \\ & ((\vee_{o\rightarrow o\rightarrow o}\, s_o)\, t_o) \mid \forall_{(\alpha\rightarrow o)\rightarrow o}(\lambda X_\alpha . s_o) \end{aligned}$$

where $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \boldsymbol{T}$. The text from Def. 2 analogously applies, except that we do not consider the modal connectives $\Box$ and $\Diamond$.

The definitions for substitution (Def. 3), β- and η-reduction (Def. 4), frame (Def. 5), and assignment (Def. 7) remain unchanged.

Definition 13. *A* model *for HOL is a tuple $\boldsymbol{M} = \langle \boldsymbol{D}, \boldsymbol{I} \rangle$, where $\boldsymbol{D}$ is a frame, and $\boldsymbol{I}$ is a family of typed interpretation functions mapping constant symbols $\boldsymbol{p_\alpha}$ to appropriate elements of $\boldsymbol{D_\alpha}$, called the* denotation of $\boldsymbol{p_\alpha}$ *(the logical connectives $\neg$, $\vee$, and $\forall$ are always given the standard denotations, see below). Moreover, we assume that the domains $\boldsymbol{D_{\alpha\to\alpha\to o}}$ contain the respective identity relations.*

Definition 14. *The* value $\|\boldsymbol{s_\alpha}\|^{\boldsymbol{M,g}}$ *of a HOL term $\boldsymbol{s_\alpha}$ on a model $\boldsymbol{M} = \langle \boldsymbol{D}, \boldsymbol{I} \rangle$ under assignment $\boldsymbol{g}$ is an element $\boldsymbol{d} \in \boldsymbol{D_\alpha}$ defined in the following way:*

1. $\|\boldsymbol{p_\alpha}\|^{\boldsymbol{M,g}} = \boldsymbol{I}(\boldsymbol{p_\alpha})$
2. $\|\boldsymbol{X_\alpha}\|^{\boldsymbol{M,g}} = \boldsymbol{g}(\boldsymbol{X_\alpha})$
3. $\|(\boldsymbol{s_{\alpha\to\beta}}\, \boldsymbol{t_\alpha})_\beta\|^{\boldsymbol{M,g}} = \|\boldsymbol{s_{\alpha\to\beta}}\|^{\boldsymbol{M,g}}(\|\boldsymbol{t_\alpha}\|^{\boldsymbol{M,g}})$
4. $\|(\boldsymbol{\lambda X_\alpha . s_\beta})_{\alpha\to\beta}\|^{\boldsymbol{M,g}}$ = *the function* $\boldsymbol{f}$ *from* $\boldsymbol{D_\alpha}$ *to* $\boldsymbol{D_\beta}$ *such that* $\boldsymbol{f(d)} = \|\boldsymbol{s_\beta}\|^{\boldsymbol{M,g[d/X_\alpha]}}$ *for all* $\boldsymbol{d} \in \boldsymbol{D_\alpha}$
5. $\|(\neg_{\boldsymbol{o\to o}}\, \boldsymbol{s_o})_o\|^{\boldsymbol{M,g}} = \boldsymbol{T}$ *if and only if* $\|\boldsymbol{s_o}\|^{\boldsymbol{M,g}} = \boldsymbol{F}$
6. $\|((\vee_{\boldsymbol{o\to o\to o}}\, \boldsymbol{s_o})\, \boldsymbol{t_o})_o\|^{\boldsymbol{M,g}} = \boldsymbol{T}$ *if and only if* $\|\boldsymbol{s_o}\|^{\boldsymbol{M,g}} = \boldsymbol{T}$ *or* $\|\boldsymbol{t_o}\|^{\boldsymbol{M,g}} = \boldsymbol{T}$
7. $\|(\forall_{\boldsymbol{(\alpha\to o)\to o}}(\boldsymbol{\lambda X_\alpha . s_o}))_o\|^{\boldsymbol{M,g}} = \boldsymbol{T}$ *if and only if for all* $\boldsymbol{d} \in \boldsymbol{D_\alpha}$ *we have* $\|\boldsymbol{s_o}\|^{\boldsymbol{M,g[d/X_\alpha]}} = \boldsymbol{T}$

The definition for standard and Henkin models (Def. 9), and for truth in a model, validity, etc. (Def. 10) are adapted in the obvious way, and we use the notation $\boldsymbol{M, g} \models \boldsymbol{s_o}$, $\models \boldsymbol{s_o}$. Moreover, we write $\boldsymbol{\Gamma} \models \boldsymbol{\Delta}$ (for sets of formulas Γ and Δ) if and only if there is a model $\boldsymbol{M} = \langle \boldsymbol{D}, \boldsymbol{I} \rangle$ and an assignment g such that $M, g \models s_o$ for all $s_o \in \Gamma$ and $M, g \models t_o$ for at least one $t_o \in \Delta$. As for HOML, we assume Henkin semantics in the remainder.

3.2 Semantic Embedding

Before we now present the embedding of HOML in HOL a clarifying remark concerning flexible and rigid constant symbols is required.

Remark 1. In Definition 6, constants are assumed to be *flexible*, because their interpretations may depend on worlds. A constant p_α is said to be *rigid* if it has the same interpretation in all worlds (i.e. there exists $d \in D_\alpha$ such that for all worlds w, $I_w(p_\alpha) = d$). For the sake of simplicity, we assume from now on (except in Section 5.4) that for every type α different from o, all constant symbols p_α are rigid. With this assumption, we may work with a non-world-indexed interpretation function I for types different from o. Clearly, I is then chosen so that $I(p_\alpha) = I_w(p_\alpha)$ for all w and for all p_α.

The encoding of HOML in HOL is simple: we identify HOML formulas of type o with certain HOL formulas (predicates) of type $\boldsymbol{\iota \to o}$. The HOL type $\boldsymbol{\iota \to o}$ is abbreviated in the remainder as $\boldsymbol{\sigma}$.

Definition 15. *We define for each HOML type $\alpha \in T$ the associated raised HOL type $\lceil\alpha\rceil$ as follows:*

$$\begin{aligned}\lceil\mu\rceil &= \boldsymbol{\mu}\\ \lceil o\rceil &= \boldsymbol{\sigma} = \boldsymbol{\iota}\to\boldsymbol{o}\\ \lceil\alpha\to\beta\rceil &= \lceil\alpha\rceil\to\lceil\beta\rceil\end{aligned}$$

Hence, all HOML terms are rigid, except for those of type o.

Definition 16. *HOML terms s_α are associated with type-raised HOL terms $\lceil s_\alpha\rceil$ in the following way:*

$$\begin{aligned}\lceil p_\alpha\rceil &= \boldsymbol{p}_{\lceil\alpha\rceil}\\ \lceil X_\alpha\rceil &= \boldsymbol{X}_{\lceil\alpha\rceil}\\ \lceil(s_{\alpha\to\beta}\,t_\alpha)\rceil &= (\lceil s_{\alpha\to\beta}\rceil\,\lceil t_\alpha\rceil)\\ \lceil(\lambda X_\alpha.\,s_\beta)\rceil &= (\boldsymbol{\lambda}\lceil X_\alpha\rceil.\lceil s_\beta\rceil)\\ \lceil(\neg_{o\to o}\,s_o)\rceil &= (\dot\neg_{\boldsymbol{\sigma}\to\boldsymbol{\sigma}}\,\lceil s_\alpha\rceil)\\ \lceil((\vee_{o\to o\to o}\,s_o)\,t_o)\rceil &= ((\dot\vee_{\boldsymbol{\sigma}\to\boldsymbol{\sigma}\to\boldsymbol{\sigma}}\,\lceil s_\alpha\rceil)\,\lceil t_\alpha\rceil)\\ \lceil((\forall_{(\alpha\to o)\to o}\,(\lambda X_\alpha.\,s_\beta)\rceil &= (\dot\forall_{(\boldsymbol{\alpha}\to\boldsymbol{\sigma})\to\boldsymbol{\sigma}}\,(\boldsymbol{\lambda}\lceil X_\alpha\rceil.\lceil s_\beta\rceil)\\ \lceil(\Box_{o\to o}\,s_o)\rceil &= (\dot\Box_{\boldsymbol{\sigma}\to\boldsymbol{\sigma}}\,\lceil s_\alpha\rceil)\end{aligned}$$

where $\dot\neg, \dot\vee, \dot\forall$, and $\dot\Box$ are the type-raised modal HOL connectives *associated with the corresponding modal HOML connectives. They are defined as follows (where $\boldsymbol{r}_{\boldsymbol{\iota}\to\boldsymbol{\iota}\to\boldsymbol{o}}$ is a new constant symbol in HOL associated with the accessibility relation R of HOML):*

$$\begin{aligned}\dot\neg_{\boldsymbol{\sigma}\to\boldsymbol{\sigma}} &= \boldsymbol{\lambda s_\sigma.\lambda W_\iota.\neg\,(s\,W)}\\ \dot\vee_{\boldsymbol{\sigma}\to\boldsymbol{\sigma}\to\boldsymbol{\sigma}} &= \boldsymbol{\lambda s_\sigma.\lambda t_\sigma.\lambda W_\iota.\,s\,W\vee t\,W}\\ \dot\forall_{(\boldsymbol{\alpha}\to\boldsymbol{\sigma})\to\boldsymbol{\sigma}} &= \boldsymbol{\lambda s_{\alpha\to\sigma}.\lambda W_\iota.\forall X_\alpha.\,s\,X\,W}\\ \dot\Box_{\boldsymbol{\sigma}\to\boldsymbol{\sigma}} &= \boldsymbol{\lambda s_\sigma.\lambda W_\iota.\forall V_\iota.\neg(r_{\iota\to\iota\to o}\,W\,V)\vee s\,V}\end{aligned}$$

As before, we write $\dot\forall\boldsymbol{X_\alpha.s_\sigma}$ as shorthand for $\dot\forall_{(\boldsymbol{\alpha}\to\boldsymbol{\sigma})\to\boldsymbol{\sigma}}(\boldsymbol{\lambda X_\alpha.s_\sigma})$. Further operators, such as $\dot\top$, $\dot\bot$, $\dot\wedge$, $\dot\supset$, $\dot\equiv$, $\dot\Diamond$, and $\dot\exists$ ($\dot\exists\boldsymbol{X_\alpha.s_\sigma}$ is used as shorthand for $\dot\exists_{(\boldsymbol{\alpha}\to\boldsymbol{\sigma})\to\boldsymbol{\sigma}}(\boldsymbol{\lambda X_\alpha.s_\sigma})$) can now be easily defined. Moreover, we can define further modal operators, such as the difference modality $\boldsymbol{D}$, the global modality $\boldsymbol{E}$, nominals with $\boldsymbol{!}$, and the $\boldsymbol{@}$ operator (cf. [21]). The above equations can be treated as abbreviations in HOL theorem provers. Alternatively, they can be stated as axioms where $=$ is either Leibniz equality or primitive equality (if additionally provided in the HOL grammar, as is the case for most modern HOL provers).

As a consequence of the above embedding we can express HOML proof problems elegantly in the type-raised syntax of HOL. By rewriting or expanding definitions, we can reduce these representations to corresponding statements containing only the basic HOL connectives $\neg_{\boldsymbol{o}\to\boldsymbol{o}}$, $\vee_{\boldsymbol{o}\to\boldsymbol{o}\to\boldsymbol{o}}$, and $\forall_{(\boldsymbol{\alpha}\to\boldsymbol{o})\to\boldsymbol{o}}$.

Example 1. The HOML formula $\Box\,\exists P_{\mu\to o}.\,P\,a_\mu$ is associated with the type raised HOL formula $\boldsymbol{\dot{\Box}\,\dot{\exists}P_{\mu\to\sigma}.\,P\,a_\mu}$, which rewrites into the following $\beta\eta$-normal HOL term of type $\boldsymbol{\sigma}$

$$\boldsymbol{\lambda W_\iota.\forall V_\iota.\neg(r\,W\,V)\vee\neg\forall P_{\mu\to\sigma}.\neg(P\,a_\mu\,V)}$$

Next, we define validity of type-raised modal HOL propositions $\boldsymbol{s_\sigma}$ in the obvious way: $\boldsymbol{s_\sigma}$ is valid if and only if for all possible worlds $\boldsymbol{w_\iota}$ we have $\boldsymbol{w_\iota\in s_\sigma}$, that is, if and only if $\boldsymbol{(s_\sigma\,w_\iota)}$ holds.

Definition 17. Validity *is modeled as an abbreviation for the following λ-term:*

$$\boldsymbol{valid}=\boldsymbol{\lambda s_{\iota\to o}.\forall W_\iota.s\,W}$$

(alternatively, we could define validity simply as $\forall_{(\iota\to o)\to o}$). Instead of $\boldsymbol{valid\ s_\sigma}$ *we also use the notation* $[\boldsymbol{s_\sigma}]$.

Example 2. We analyze whether the type-raised modal HOL formula $\boldsymbol{\dot{\Box}\,\dot{\exists}P_{\mu\to\sigma}.(P\,a_\mu)}$ is valid or not. For this, we formalize the HOL proof problem $[\boldsymbol{\dot{\Box}\,\dot{\exists}P_{\mu\to\sigma}.(P\,a_\mu)}]$, which expands into $\boldsymbol{\forall W_\iota.\forall V_\iota.\neg(r\,W\,V)\vee\neg\forall P_{\mu\to\sigma}.\neg(P\,a_\mu\,V)}$. It is easy to check that this term is valid in Henkin semantics: put $\boldsymbol{P=\lambda X_\mu.\lambda Y_\iota.\top}$.

3.3 Soundness and Completeness

Theorem 1 (Soundness and Completeness). *For all HOML formulas s_o we have:*

$$\models s_o \quad \textit{if and only if} \quad \models [\lceil s_o\rceil]$$

Proof sketch: *The proof adapts the ideas presented in [21]. By contraposition it is sufficient to show $\not\models s_o$ if and only if $\not\models [\lceil s_o\rceil]$, that is, $\|s_o\|^{M,g,w}$ (for some HOML model M, assignment g, and w) if and only if $\|\forall\boldsymbol{W_\iota.\lceil s_o\rceil\,W}\|^{\boldsymbol{M,g}}$ (for some HOL model $\boldsymbol{M}$ and assignment $\boldsymbol{g}$) if and only if $\|\boldsymbol{\lceil s_o\rceil\,W}\|^{\boldsymbol{M,g[w/W]}}$ (for some $\boldsymbol{M}$, $\boldsymbol{g}$, and w). We easily get the proof by choosing the obvious correspondences between D and $\boldsymbol{D}$, W and $\boldsymbol{D_\iota}$, I and $\boldsymbol{I}$, g and $\boldsymbol{g}$, R and $r_{\iota\to\iota\to o}$, and w and $\boldsymbol{w}$.* □

From Theorem 1 we, for example, get the following corollaries:

$$\models^{KB} s_o \quad \text{if and only if} \quad \boldsymbol{(\textbf{symmetric}\,r_{\iota\to\iota\to o})}\models[\lceil s_o\rceil]$$

$$\models^{S5} s_o \quad \text{if and only if} \quad \boldsymbol{(\textbf{equiv-rel}\,r_{\iota\to\iota\to o})}\models[\lceil s_o\rceil]$$

where **symmetric** and **equiv-rel** are defined in an obvious way. Analogous corollaries can be stated for other normal modal logics including, for example, KD and S4.

3.4 Logic Variations

The semantics of a higher-order modal logic depends on subtle and often implicit assumptions. In the following two subsubsections, we explicitly discuss which assumptions have been made in the previous sections and how different choices would lead to different higher-order modal logics.

Constant, Varying and Cumulative Domains

In the previous sections we have focused on quantification over *constant domains*, which assumes that all individuals in D_μ actually exist in all worlds. Alternatively, quantified modal logics may also use quantification over *varying domains*, which assumes that the subset of individuals actually existing in a world w may depend on w.

Techniques for handling varying domain quantification in the embedding of first-order modal logics in HOL have been outlined in [17], and they can be extended to higher-order modal logics as well. For this, the following modifications are required:

1. The definition of $\dot{\forall}$ (for type $(\mu \to \sigma) \to \sigma$, which encodes first-order quantification, is modified as follows: $\dot{\forall} = \boldsymbol{\lambda s_{\mu\to\sigma}.\lambda w_\iota.\forall x_\mu.}\mathbf{ExistsInW}\, \boldsymbol{x\, w} \supset \boldsymbol{s\, x\, w}$, where the relation $\mathbf{ExistsInW}_{\boldsymbol{\mu\to\iota\to o}}$ (for 'Exists in world') relates individuals with worlds. The sets $\boldsymbol{\{x \mid}\mathbf{ExistsInW}\, \boldsymbol{x\, w\}}$ are the possibly varying individual domains associated with the worlds $\boldsymbol{w}$.
2. A non-emptiness axiom for these individual domains is added: $\boldsymbol{\forall w_\iota \exists x_\mu}\mathbf{ExistsInW}\, \boldsymbol{x\, w}$
3. For each individual constant symbol $\boldsymbol{c}$ in the proof problem an axiom $\boldsymbol{\forall w_\iota.}\mathbf{ExistsInW}\, \boldsymbol{c\, w}$ is postulated; these axioms enforce the designation of $\boldsymbol{c}$ in the individual domain of each world $\boldsymbol{w}$. Analogous designation axioms are required for function symbols.

Modifications 1–3 adapt the HOL approach to varying domains. For the special case of *cumulative domains*, in which the varying domains are assumed to be increasing along the accessibility relation $\boldsymbol{r}$, an additional modification is needed:

4. The axiom $\boldsymbol{\forall x_\mu.\forall v_\iota.\forall w_\iota.}\mathbf{ExistsInW}\, \boldsymbol{x\, v} \wedge \boldsymbol{r\, v\, w} \Rightarrow \mathbf{ExistsInW}\, \boldsymbol{x\, w}$ is added.

If we were using a richer higher-order logic with not only simple types but also dependent types, we could achieve varying domains without using existence predicates, by making the type of individuals depend on worlds.

Rigidity and Flexibility

In the previous sections, it is assumed that all terms (except terms of boolean type) are *rigid*: independent of the world. The alternative option of *flexible* terms can be easily handled by type-raising. For example, a flexible HOML constant symbol $\mathrm{kingOfFrance}_\mu$ would be mapped to a type-raised (and thus world-dependent) HOL constant symbol $\mathbf{kingOfFrance}_{\iota\to\mu}$. Higher-order modal logics with flexible terms may, for example, be useful for dealing with certain kinds of inconsistency, as discussed in Section 5.4.

4 Reasoning Tools for Higher-Order Modal Logic

The above approach to automate HOML in HOL can be employed in combination with any ATP system that is sound and (possibly) complete for HOL with Henkin semantics. The embeddings approach is particularly simple to implement, because it does not require any modification in the source code of the HOL prover.

4.1 TPTP `thf0`-compliant Reasoning Tools

An encoding of second-order modal logic KB in HOL using the concrete TPTP `thf0`-syntax[3] [72] is exemplarily provided in Figure 1.[4] The lifted modal connectives $\dot{\neg}, \dot{\vee}, \dot{\wedge}, \dot{\supset}, \dot{\Box}, \dot{\Diamond}, \dot{\forall}$ and $\dot{\exists}$ are in this representation called `mnot, mor, mand, mimplies, mbox, mdia, mforall` and `mexists`. Since `thf0` does not support polymorphism, a generic modeling of `mforall` and `mexists` is not possible here and concrete instances of these quantifiers for individuals and sets of individuals (properties) are provided instead. Of course, further copies of these definitions could be added and adapted in order to obtain quantifiers for higher-types.

The given set of axioms turns any `thf0`-compliant HOL-ATP in a reasoning tool for second-order modal logic KB. Examples for `thf0`-compliant provers are LEO-II [12], Satallax [30], Isabelle [54], agsyHOL [50], HOLyHammer [48], cocATP and Nitpick [28]. Nitpick is specialized in (counter-)model finding. The other systems are in the first place theorem provers, although Satallax and LEO-II may occasionally also find countermodels for given non-valid conjectures.

The `thf0`-encoding from Figure 1 has been applied and tested with the provers LEO-II, Satallax and Nitpick in the context of our work on the ontological argument for the existence of God [19]; more on this study will be provided in Section 5.3. Figure 2 presents a most prominent proof problem from these studies in `thf0`-syntax. In Figure 2 an improved (but more spacious) formatting is employed; such a formatting can easily be obtained with the help of the TPTP2X or TPTP4X tools of Sutcliffe's SystemOnTPTP infrastructure [71].

In the context of first-order modal logic (FML) theorem proving, the FMLtoHOL tool [23] has been developed, which converts problems in FML, formulated in `qmf`-syntax [64] (which extends the TPTP `fol`-syntax [71] with operators `#box` and `#dia`), into HOL problems in `thf0`-syntax. FMLtoHOL automatically transforms constant domain FML problems in corresponding HOL problems [21]. The tool has been extended to also support varying and cumulative domains. At present FMLtoHOL supports modal logics from $L := \{\text{K,K4,D,D4,T,S4,S5}\}$.

The FMLtoHOL tool has been exemplarily applied in combination with a meta-prover for HOL. This meta-prover exploits the SystemOnTPTP infrastructure [71] and sequentially schedules the HOL reasoners LEO-II, Satallax, Isabelle, agsyHOL and Nitpick. The system has been evaluated with respect to 580 benchmark problems in the QMLTP library [64]. As a side contribution, the complete translation of the QMLTP

[3] `thf` stands for *typed higher-order form* and it refers to a family of syntax formats for higher-order logic. So far only the fully developed `thf0` format, for simple type theory, is in practical use.

[4] In `thf0`, which is a concrete syntax for HOL, `$i` and `$o` represent the HOL base types i and o (Booleans). `$i>$o` encodes a function (predicate) type. Predicate application, as in $A(X,W)$, is encoded as `((A@X)@W)` or simply as `(A@X@W)`, i.e., function/predicate application is represented by `@`; universal quantification and λ-abstraction as in $\lambda A_{i\rightarrow o} \forall W_i (A\,W)$ and are represented as in `^[X:$i>$o]:![W:$i]:(A@W)`; comments begin with `%`.

```
1  %----The base type $i (already built-in) stands here for worlds and
2  %----mu for individuals; $o (also built-in) is the type of Booleans
3  thf(mu_type,type,(mu:$tType)).
4  %----Reserved constant r for accessibility relation
5  thf(r,type,(r:$i>$i>$o)).
6  %----Modal logic operators not, or, and, implies, box, diamond
7  thf(mnot_type,type,(mnot:($i>$o)>$i>$o)).
8  thf(mnot,definition,(mnot = (^[A:$i>$o,W:$i]:~(A@W)))).
9  thf(mor_type,type,(mor:($i>$o)>($i>$o)>$i>$o)).
10 thf(mor,definition,(mor = (^[A:$i>$o,Psi:$i>$o,W:$i]:((A@W)|(Psi@W))))).
11 thf(mand_type,type,(mand:($i>$o)>($i>$o)>$i>$o)).
12 thf(mand,definition,(mand = (^[A:$i>$o,Psi:$i>$o,W:$i]:((A@W)&(Psi@W))))).
13 thf(mimplies_type,type,(mimplies:($i>$o)>($i>$o)>$i>$o)).
14 thf(mimplies,definition,(
15     mimplies = (^[A:$i>$o,Psi:$i>$o,W:$i]:((A@W)&(Psi@W))))).
16 thf(mbox_type,type,(mbox:($i>$i>$o)>($i>$o)>$i>$o)).
17 thf(mbox,definition,(mbox = (^[A:$i>$o,W:$i]:![V:$i]:(~(r@W@V)|(A@V))))).
18 thf(mdia_type,type,(mdia:($i>$i>$o)>($i>$o)>$i>$o)).
19 thf(mdia,definition,(mdia = (^[A:$i>$o,W:$i]:?[V:$i]:((r@W@V)&(A@V))))).
20 %----Quantifiers (constant domains) for individuals and propositions
21 thf(mforall_ind_type,type,(mforall_ind:(mu>$i>$o)>$i>$o)).
22 thf(mforall_ind,definition,(
23     mforall_ind = (^[A:mu>$i>$o,W:$i]:![X:mu]:(A@X@W)))).
24 thf(mforall_indset_type,type,(mforall_indset:((mu>$i>$o)>$i>$o)>$i>$o)).
25 thf(mforall_indset,definition,(
26     mforall_indset = (^[A:(mu>$i>$o)>$i>$o,W:$i]:![X:mu>$i>$o]:(A@X@W)))).
27 thf(mexists_ind_type,type,(mexists_ind:(mu>$i>$o)>$i>$o)).
28 thf(mexists_ind,definition,(
29     mexists_ind = (^[A:mu>$i>$o,W:$i]:?[X:mu]:(A@X@W)))).
30 thf(mexists_indset_type,type,(mexists_indset:((mu>$i>$o)>$i>$o)>$i>$o)).
31 thf(mexists_indset,definition,(
32     mexists_indset = (^[A:(mu>$i>$o)>$i>$o,W:$i]:?[X:mu>$i>$o]:(A@X@W)))).
33 %----Definition of validity (grounding of lifted modal formulas)
34 thf(v_type,type,(v:($i>$o)>$o)).
35 thf(mvalid,definition,(v = (^[A:$i>$o]:![W:$i]:(A@W)))).
36 %----Properties of accessibility relations: symmetry
37 thf(msymmetric_type,type,(msymmetric:($i>$i>$o)>$o)).
38 thf(msymmetric,definition,(
39     msymmetric = (^[R:$i>$i>$o]:![S:$i,T:$i]:((R@S@T)=>(R@T@S))))).
40 %----Here we work with logic KB, i.e., we postulate symmetry for r
41 thf(sym,axiom,(msymmetric@r)).
```

Fig. 1. HOL encoding of second-order modal logic KB in `thf0`-syntax. Modal formulas are mapped to HOL predicates (with type `$i>$o`); type `$i` now stands for possible worlds. The modal connectives $\neg$ (`mnot`), $\vee$ (`mor`) and $\Box$ (`mbox`), universal quantification for individuals (`mall_ind`) and for sets of individuals (`mall_indset`) are introduced in lines 7-18. Validity of lifted modal formulas is defined in the standard way (lines 20-21). Symmetry of accessibility relation r is postulated in lines 23-26. Hence, second-order KB is realized here; for logic K the symmetry axiom can be dropped.

library (for all logics in L, all different domain conditions, and both options as explained in (C)) into HOL (resp. `thf0`) resulted in $7 \times 3 \times 2 \times 580 = 24360$ new problems.[5]

Experiments [23] show that the FMLtoHOL approach to automate FMLs is very competitive. Regarding the combined performance (number of proved or refuted problems) the HOL approach performed best in this study.

[5] The 3480 problems for logic S4 can be download from `http://christoph-benzmueller.de/papers/THF-S4-ALL.zip`.

```
%------------------------------------------------------------------------------
%----Axioms for Quantified Modal Logic KB.
include('Quantified_KB.ax').
%------------------------------------------------------------------------------
%----constant symbol for positive: p
thf(p_tp,type,( p: ( mu > $i > $o ) > $i > $o )).

%----constant symbol for God-like: g
thf(g_tp,type,( g: mu > $i > $o )).

%----constant symbol for essence: ess
thf(ess_tp,type,( ess: ( mu > $i > $o ) > mu > $i > $o )).

%----constant symbol for necessary existence: ne
thf(ne_tp,type,( ne: mu > $i > $o )).

%----D1: A God-like being possesses all positive properties.
thf(defD1,definition, (
    g = ( ^ [X: mu] :
            ( mforall_indset
            @ ^ [Phi: mu > $i > $o] :
                ( mimplies @ ( p @ Phi ) @ ( Phi @ X ) ) ) ) )).

%----C: Possibly, God exists. (Proved before)
thf(corC,axiom,
    ( v
    @ ( mdia
      @ ( mexists_ind
        @ ^ [X: mu] :
            ( g @ X ) ) ) )).

%----T2: Being God-like is an essence of any God-like being. (Proved before)
thf(thmT2,axiom,
    ( v
    @ ( mforall_ind
      @ ^ [X: mu] :
          ( mimplies @ ( g @ X ) @ ( ess @ g @ X ) ) ) )).

%----D3: Necessary existence of an individual is the necessary
%----exemplification of all its essences
thf(defD3,definition,(
    ne = ( ^ [X: mu] :
             ( mforall_indset
             @ ^ [Phi: mu > $i > $o] :
                 ( mimplies @ ( ess @ Phi @ X )
                 @ ( mbox
                   @ ( mexists_ind
                     @ ^ [Y: mu] :
                         ( Phi @ Y ) ) ) ) ) ) )).

%----A5: Necessary existence is positive.
thf(axA5,axiom, ( v @ ( p @ ne ) )).

%----T3: Necessarily God exists.
thf(thmT3,conjecture,
    ( v
    @ ( mbox
      @ ( mexists_ind
        @ ^ [X: mu] :
            ( g @ X ) ) ) )).

```

Fig. 2. TPTP `thf0`-encoding of theorem T3 in Scott's adaptation (see also Figure 1) of Gödel's ontological argument [19, 66].

4.2 Interactive Proof Assistants – `Isabelle`

The TPTP THF embedding of HOML is very useful for flexible proof automation of HOML with off-the-shelf HOL-ATPs. Unfortunately, however, it is not particularly well suited for enabling user interaction at an intuitive abstraction level. In this subsection we therefore briefly illustrate how the embedding of HOML can be encoded and exploited in the interactive proof assistant Isabelle/HOL. A very useful tool of Isabelle/HOL is Sledgehammer [27], which connects the Isabelle core system with external ATPs, including remote calls to the LEO-II and Satallax provers running at the SystemOnTPTP infrastructure in Miami.

An embedding of HOML with constant domain semantics in Isabelle/HOL is presented in the upper part of Figure 3, which displays the content of an Isabelle theory file named `QML.thy`.[6] Note that in the definition of `mforall` and `mexists` a type variable `'a` is used. Thus, in contrast to the non-polymorphic TPTP THF encoding of second-order modal logic from above, polymoprhic quantifiers are introduced here to obtain full HOML. Additional quantifiers for varying domains can easily be added, this is illustrated in the lower part of Figure 3.[7] An obvious advantage of Isabelle is its comparably good notation support in the user interface. The connectives `mnot, mor, mand, mimplies, mbox, mdia, mforall` and `mexists` are displayed here as m$\neg$, m$\vee$, m$\wedge$, m$\rightarrow$, $\Box$, $\Diamond$, $\forall$ and $\exists$.

Figure 4 exemplarily displays the development of Gödel's ontological argument (in Scott's version, cf. Figure 11) in Isabelle/HOL. Varying domain quantifiers for individuals are employed in this particular encoding; see e.g. the occurrence of $\forall e$ in Axiom `A2` and the occurrence of $\exists e$ in Theorem `T3`. Note that the second-order quantifier $\forall$, as used for instance in `T1`, is a constant domain quantifier. Hence, we here illustrate the flexibility of the embeddings approach, in which we can even easily mix different types of quantifiers. Note that proofs in Figure 4 are fully automatic; here Isabelle's Metis prover is used. However, Metis has to be called here with the appropriate assumptions. When using Sledgehammer instead, for example, in combination with LEO-II, Satallax or other ATPs, the respective assumptions can be avoided in the Sledgehammer call and will be automatically determined.

4.3 Interactive Proof Assistants – `Coq`

We have already seen how the embedding approach is flexible and effective for *fully automated* reasoning. However, one may wonder whether the embedding approach is adequate also for intuitive *interactive* reasoning, when the user proves theorems by interacting with a proof assistant such as `Coq`. In this section, we study this question, and show that the answer is positive.

One major concern is whether the embedding could be a disturbance to the user. Fortunately, by using `Coq`'s `Ltac` tactic language, we are able to define intuitive new tactics that hide the technical details of the embedding from the user. The resulting infrastructure for modal reasoning within `Coq` provides a user experience where modalities

[6] See file `QML.thy` available at `https://github.com/FormalTheology/GoedelGod/blob/master/Formalizations/Isabelle/`

[7] See file `QML_var.thy` at the github url from above.

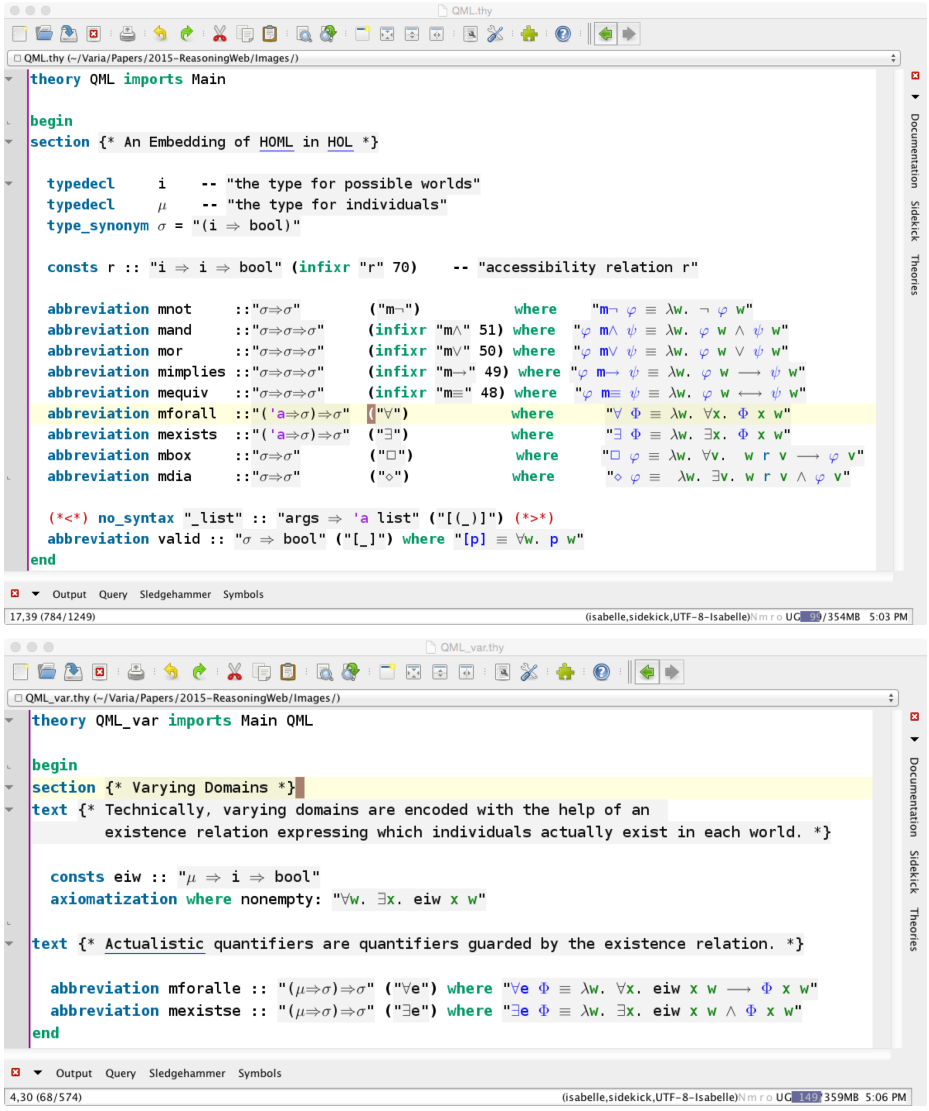


Fig. 3. Isabelle/HOL embedding of HOML K (above), and the subsequent extension of this theory by varying domain quantifiers ∀e and ∃e for individuals.

can be handled transparently and straightforwardly. Therefore, a user with basic knowledge of modal logics and Coq's tactics should be able to use (and extend) our implementation with no excessive overhead. It should be straightforward to analogously implement respective tactics in other interactive proof assistants, including Isabelle/HOL.

As before, the first step in the shallow embedding of modal logics is the declaration of a type for worlds. Modal propositions are then not of type Prop but of a lifted type

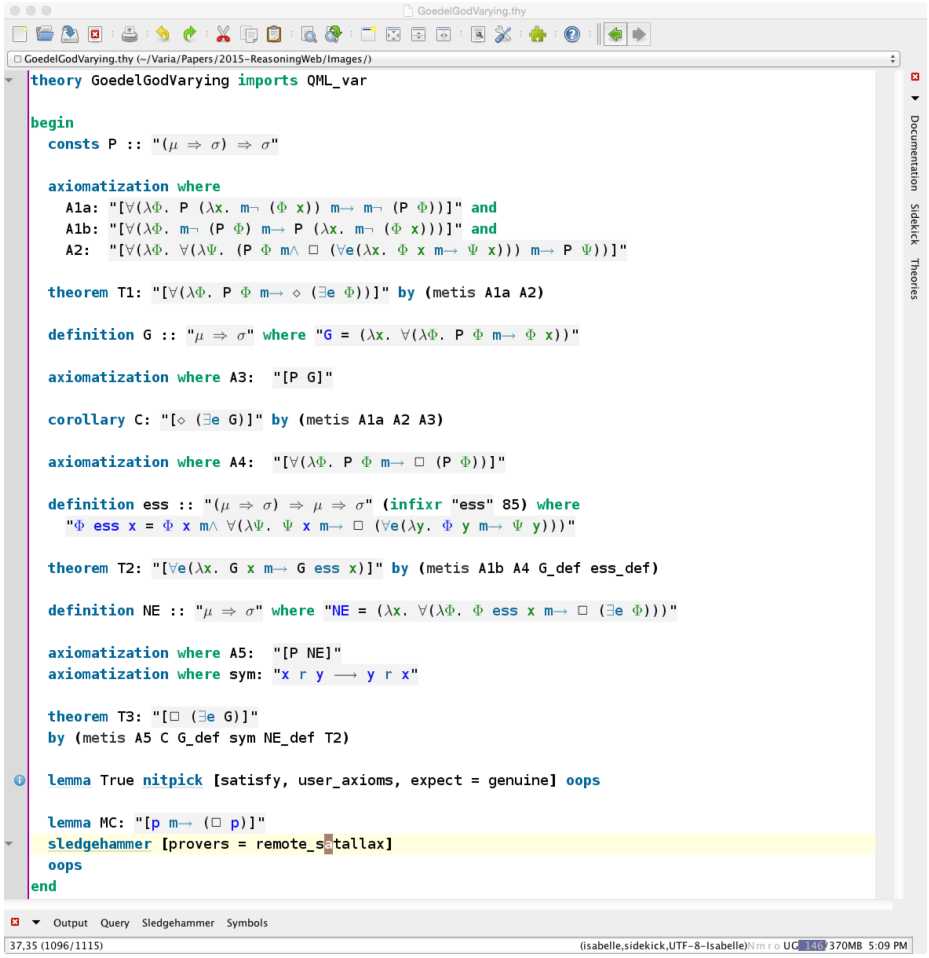


Fig. 4. Scott's version of Gödel's ontological argument encoded and proved in Isabelle/HOL. Varying domain quantifiers for individuals are mixed with constant domain quantifiers for properties of individuals.

o that depends on possible worlds (o corresponds to σ in the Isabelle/HOL encoding from before):

```
Parameter i: Type. (* Type for worlds *)
Parameter u: Type. (* Type for individuals *)
Definition o := i -> Prop. (* Type of modal propositions *)
```

Possible worlds are connected by an accessibility relation, which can be represented in Coq by a parameter r, as follows:

```
Parameter r: i -> i -> Prop. (* Accessibility relation for worlds *)
```

As before, all modal connectives are simply lifted versions of the usual logical connectives. Notations are used to allow the modal connectives to be used as similarly as

possible to the usual connectives. As before, the prefix "m" is used to distinguish the modal connectives: if $\odot$ is a connective on type Prop, m$\odot$ is a connective on the lifted type o of modal propositions.

```
Definition mnot (p: o)(w: i) := ~ (p w).
Notation "m~  p" := (mnot p) (at level 74, right associativity).

Definition mand (p q:o)(w: i) := (p w) /\ (q w).
Notation "p m/\ q" := (mand p q) (at level 79, right associativity).

Definition mor (p q:o)(w: i) := (p w) \/ (q w).
Notation "p m\/ q" := (mor p q) (at level 79, right associativity).

Definition mimplies (p q:o)(w:i) := (p w) -> (q w).
Notation "p m-> q" := (mimplies p q) (at level 99, right associativity).

Definition mequiv (p q:o)(w:i) := (p w) <-> (q w).
Notation "p m<-> q" := (mequiv p q) (at level 99, right associativity).

Definition mequal (x y: u)(w: i) := x = y.
Notation "x m= y" := (mequal x y) (at level 99, right associativity).
```

Likewise, modal quantifiers are lifted versions of the usual quantifiers. Coq's type system with dependent types is particularly helpful here. The modal quantifiers A and E are defined as depending on a type t. Therefore, they can quantify over variables of any type. Moreover, the curly brackets indicate that t is an implicit argument that can be inferred by Coq's type inference mechanism. This allows notations[8] (i.e. mforall and mexists) that mimic the notations for Coq's usual quantifiers (i.e. forall and exists).

```
Definition A {t: Type}(p: t -> o)(w: i) := forall x, p x w.
Notation "'mforall'  x , p" := (A (fun x => p))
  (at level 200, x ident, right associativity) : type_scope.
Notation "'mforall' x : t , p" := (A (fun x:t => p))
  (at level 200, x ident, right associativity,
    format "'[' 'mforall' '/ '  x  :  t , '/ '  p ']'")
  : type_scope.

Definition E {t: Type}(p: t -> o)(w: i) := exists x, p x w.
Notation "'mexists' x , p" := (E (fun x => p))
  (at level 200, x ident, right associativity) : type_scope.
Notation "'mexists' x : t , p" := (E (fun x:t => p))
  (at level 200, x ident, right associativity,
    format "'[' 'mexists' '/ '  x  :  t , '/ '  p ']'")
  : type_scope.
```

The modal operators $\Diamond$ (*possibly*) and $\Box$ (*necessarily*) are defined accordingly to their meanings in the possible world semantics. $\Box p$ holds at a world w iff p holds in every world w_1 reachable from w. $\Diamond p$ holds at world w iff p holds in some world w_1 reachable from w.

```
Definition box (p: o) := fun w => forall w1, (r w w1) -> (p w1).
Definition dia (p: o) := fun w => exists w1, (r w w1) /\ (p w1).
```

A modal proposition is valid iff it holds in every possible world. This notion of modal validity is encoded by the following defined predicate:

[8] The keyword fun indicates a lambda abstraction: fun x => p (or fun x:t => p) denotes the function $\lambda x : t.p$, which takes an argument x (of type t) and returns p.

```
Definition V (p: o) := forall w, p w.
```

To prove a modal proposition `p` (of type `o`) within `Coq`, the proposition `(V p)` (of type `Prop`) should be proved instead. To increase the transparency of the embedding to the user, the following notation is provided, allowing `[ p ]` to be written instead of `(V p)`.

```
Notation "[ p ]" := (V p).
```

Interactive theorem proving in `Coq`, and likewise in other interactive proof assistants, is usually done with tactics, imperative commands that reduce the theorem to be proven (i.e. the goal) to simpler subgoals, in a bottom-up manner. The simplest tactics can be regarded as rules of a natural deduction calculus[9] (e.g. as those shown in Figure 4.3). For example: the `intro` tactic can be used to apply the introduction rules for implication and for the universal quantifier; the `apply` tactic corresponds to the elimination rules for implication and for the universal quantifier; `split` performs conjunction introduction; `exists` can be used for existential quantifier introduction and `destruct` for its elimination.

To maximally preserve user intuition in interactive modal logic theorem proving, the embedding via the possible world semantics should be as transparent as possible to the user. Fortunately, the basic Coq tactics described above automatically unfold the shallowest modal definition in the goal. Therefore, they can be used with modal connectives and quantifiers just as they are used with the usual connectives and quantifiers. The situation for the new modal operators, on the other hand, is not as simple, unfortunately.

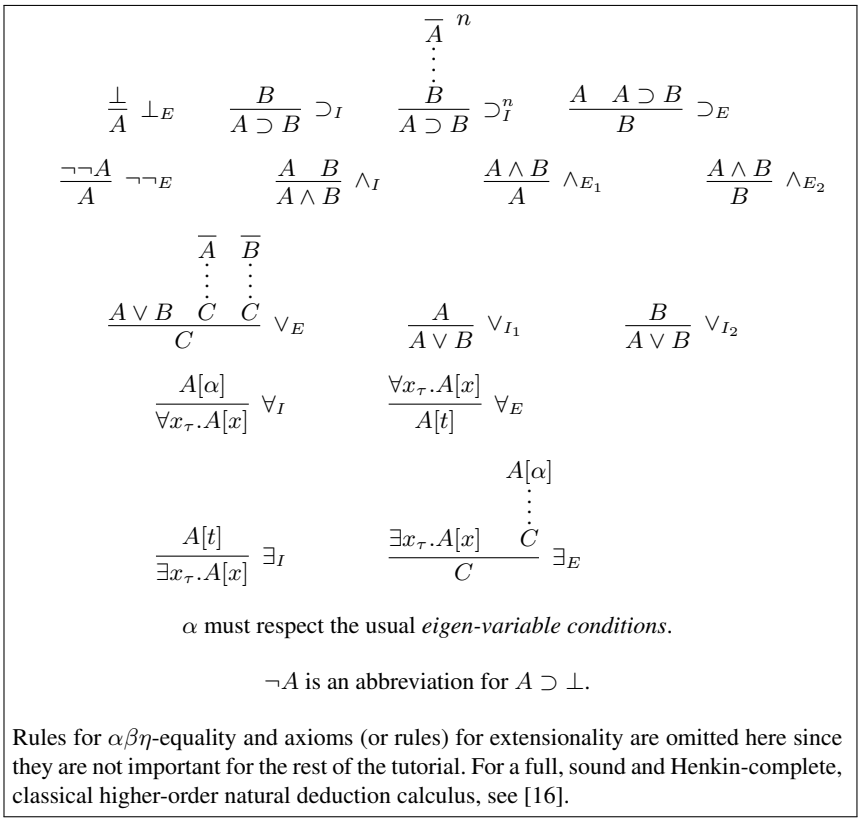
Since the modal operators are, in our embedding, essentially just abbreviations for quantifiers guarded by reachability conditions, the typical tactics for quantifiers can be used, in principle. However, this exposes the user to the technicalities of the embedding, requiring him to deal with possible worlds and their reachability explicitly. In order to obtain transparency also for the modal operators, we can implement the following specialized tactics using `Coq`'s Ltac language.

When applied to a goal of the form `((box p) w0)`, the tactic `box_i` will introduce a fresh new world `w` and then introduce the assumption that `w` is reachable from `w0`. The new goal will be `(p w)`.

```
Ltac box_i := let w := fresh "w" in let R := fresh "R"
              in (intro w at top; intro R at top).
```

If the hypothesis `H` is of the form `((box p) w0)` and the goal is of the form `(q w)`, the tactic `box_e H H1` creates a new hypothesis `H1: (p w)`. The tactic `box_elim H w1 H1` is an auxiliary tactic for `box_e`. It creates a new hypothesis `H1: (p w1)`, for any given world `w1`, not necessarily the goal's world `w`. It is also responsible for automatically trying (by `assumption`) to solve the reachability guard conditions, releasing the user from this burden.

[9] The underlying proof system of `Coq` (the Calculus of Inductive Constructions (CIC) [57]) is actually more sophisticated and minimalistic than the calculus shown in Figure 4.3. But the calculus shown here suffices for the purposes of this tutorial. This calculus is classical, because of the double negation elimination rule. Although CIC is intuitionistic, it can be made classical by importing `Coq`'s classical library, which adds the axiom of the *excluded middle* and the double negation elimination lemma.



Rules for $\alpha\beta\eta$-equality and axioms (or rules) for extensionality are omitted here since they are not important for the rest of the tutorial. For a full, sound and Henkin-complete, classical higher-order natural deduction calculus, see [16].

Fig. 5. Rules of a (classical) natural deduction calculus

```
Ltac box_elim H w1 H1 := match type of H with
      ((box ?p) ?w) => cut (p w1);
                       [intros H1 | (apply (H w1); try assumption)] end.

Ltac box_e H H1:= match goal with | [ |- (_ ?w) ] => box_elim H w H1 end.
```

If the hypothesis `H` is of the form `((dia p) w0)`, the tactic `dia_e H` generates a new hypothesis `H: (p w)` for a fresh new world `w` reachable from `w0`.

```
Ltac dia_e H := let w := fresh "w" in let R := fresh "R" in
                (destruct H as [w [R H]]; move w at top; move R at top).
```

The tactic `dia_i w` transforms a goal of the form `((dia p) w0)` into the simpler goal `(p w)` and automatically tries to solve the guard condition that `w` must be reachable from `w0`.

```
Ltac dia_i w := (exists w; split; [assumption | idtac]).
```

If the new modal tactics above are regarded from a natural deduction point of view, they correspond to the inference rules shown in Figure 4.3. Because of this correspondence

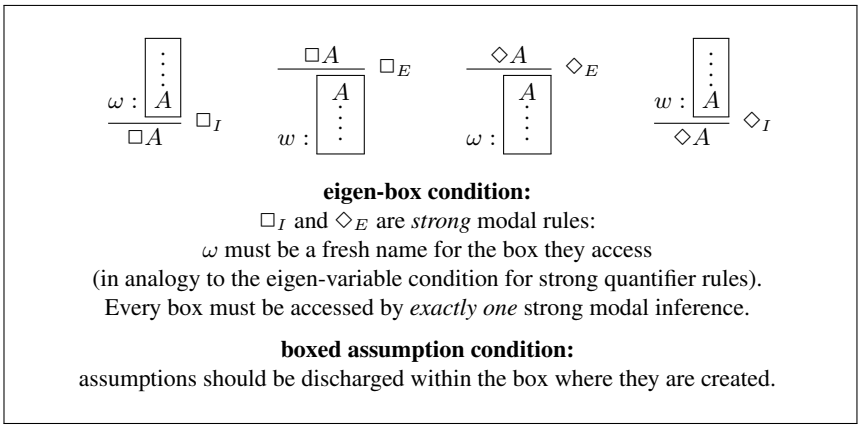


Fig. 6. Rules for modal operators

and the Henkin-completeness of the modal natural deduction calculus[10], the tactics allow the user to prove any valid modal formula without having to unfold the definitions of the modal operators.

The labels that name boxes in the inference rules of Figure 4.3 are precisely the worlds that annotate goals and hypotheses in `Coq` with the modal embedding. A hypothesis of the form `(p w)`, where `p` is a modal proposition of type `o` and `w` is a world of type `i` indicates that `p` is an assumption created inside a box with name `w`.
Finally, the tactic `mv`, standing for *modal validity*, replaces a goal of the form `[ p ]` (or equivalently `(V p)`) by a goal of the form `(p w)` for a fresh arbitrary world `w`.

```
Ltac mv := match goal with [|- (V _)] => intro end.
```

In order to illustrate the tactics described above, we show `Coq` proofs for two simple but useful modal lemmas. The first lemma resembles modus ponens, but with formulas under the scope of modal operators.

```
Lemma mp_dia:
  [mforall p, mforall q, (dia p) m-> (box (p m-> q)) m-> (dia q)].
Proof. mv.
intros p q H1 H2. dia_e H1. dia_i w0. box_e H2 H3. apply H3. exact H1.
Qed.
```

The proof of this lemma is displayed as a natural deduction proof in Figure 7. As expected, `Coq`'s basic tactics (e.g. `intros` and `apply`) work without modification. The `intros p q H1 H2` tactic application corresponds to the universal quantifier and implication introduction inferences in the bottom of the proof. The `apply H3` tactic

[10] The natural deduction calculus with the rules from Figures 4.3 and 4.3 is sound and complete relatively to the calculus of Figure 4.3 extended with a necessitation rule and the modal axiom K [68]. Starting from a sound and Henkin-complete natural deduction calculus for classical higher-order logic (cf. Figure 4.3), the additional modal rules in Figure 4.3 make it sound and Henkin-complete for the rigid higher-order modal logic **K**.

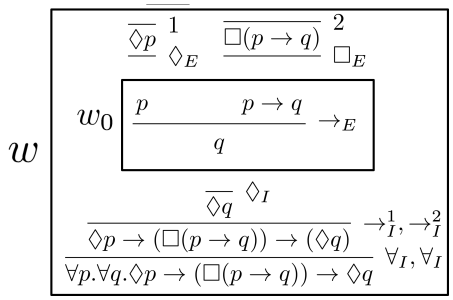


Fig. 7. Natural deduction proof of `mp_dia`

application corresponds to the implication elimination inference. The $\Diamond_E$, $\Diamond_I$ and $\Box_E$ inferences correspond, respectively, to the `dia_e H1`, `dia_i w0` and `box_e H2 H3` tactic applications. The internal box named w_0 is accessed by exactly one strong modal inference, namely $\Diamond_E$.

The same lemma could be proved without the new modal tactics, as shown below. But this is clearly disadvantageous, for several reasons: the proof script becomes longer; the definitions of modal operators must be unfolded, either explicitly (as done below) or implicitly in the user's mind; tactic applications dealing with modal operators cannot be easily distinguished from tactic applications dealing with quantifiers; and hypotheses about the reachability of worlds (e.g. `R1` below) must be handled explicitly. In summary, without the modal tactics, a convenient and intuitive correspondence between proof scripts and modal natural deduction proofs would be missing.

```
Lemma mp_dia_alternative:
  [mforall p, mforall q, (dia p) m-> (box (p m-> q)) m-> (dia q)].
Proof. mv.
intros p q H1 H2. unfold dia. unfold dia in H1. unfold box in H2.
destruct H1 as [w0 [R1 H1]]. exists w0. split.
  exact R1.
  apply H2.
    exact R1.
    exact H1.
Qed.
```

The second useful lemma allows negations to be pushed inside modalities, and again the modal tactics allow this to be proved conveniently and elegantly.

```
Lemma not_dia_box_not: [mforall p, (m~ (dia p)) m-> (box (m~ p))].
Proof. mv.
intro p. intro H. box_i. intro H2. apply H. dia_i w0. exact H2.
Qed.
```

The embedding and the new tactics allow convenient interactive reasoning for modal logic **K** within `Coq`. The axiom K is easily derivable:

```
Theorem K:
  [ mforall p, mforall q, (box (p m-> q)) m-> (box p) m-> (box q) ].
Proof. mv.
intros p q H1 H2. box_i. box_e H1 H3. apply H3. box_e H2 H4. exact H4.
Qed.
```

For other modal logics beyond **K**, their frame conditions, which constrain the reachability relation, must be stated as Coq axioms.

```
Axiom reflexivity: forall w, r w w.

Axiom transitivity: forall w1 w2 w3, (r w1 w2) -> (r w2 w3) -> (r w1 w3).

Axiom symmetry: forall w1 w2, (r w1 w2) -> (r w2 w1).
```

Hilbert-style modal logic axioms, such as for example T, can be easily derived from their corresponding frame conditions:

```
Theorem T: [ mforall p, (box p) m-> p ].
Proof. mv.
intro p. intro H. box_e H H1. exact H1. apply reflexivity.
Qed.
```

In a strong modal logic such as **S5** (which requires all three frame conditions specified above), sequences of modal operators can be collapsed to a single modal operator. One such collapsing principle is specified and proven below. By applying it iteratively, any sequence $\Diamond \ldots \Diamond \Box p$ could be collapsed to $\Box p$.

```
Theorem dia_box_to_box: [ mforall p, (dia (box p)) m-> (box p) ].
Proof. mv.
intros p H1. dia_e H1. box_i. box_e H1 H2. exact H2. eapply transitivity.
  apply symmetry. exact R.
  exact R0.
Qed.
```

It should be easily possible to analogously define corresponding HOML tactics within other interactive proof assistants, including Isabelle/HOL.

5 Applications

Propositional and quantified modal logics have (potential) applications in various fields, including, for instance, philosophy, verification, artificial intelligence agent technologies, law and linguistics (cf. [26] and the references therein). Therefore, the techniques described in these lecture notes – convenient embeddings for leveraging higher-order automated theorem provers and proof assistants for reasoning within and about modal logics – may serve as a starting point for many interesting projects, as illustrated in the following subsections.

5.1 Description Logics

Given that the embeddings approach can handle higher-order modal logics it is not surprising that the approach is also applicable to prominent description logics. For example, an Isabelle/HOL embedding of the prominent description logic ALC [11], see Figure 8, is presented in Figure 9. Note in particular the close correspondence between the embeddings of the ALC connectives and their corresponding semantical characterisations in Figure 8.

Syntax	Semantics	Description	Example
A	$A^I \subseteq \Delta^I$	atomic concept	Human, Female, ...
r	$r^I \subseteq \Delta^I \times \Delta^I$	binary relation	married, ...
$\bot$	$\emptyset$	empty concept	
$\top$	Δ^I	universal concept	
$\sim A$	$\Delta^I \setminus A^I$	complement	$\sim$ Female
$A \sqcup B$	$A^I \cup B^I$	disjunktion	Female $\sqcup$ Male
$A \sqcap B$	$A^I \cap B^I$	conjunction	Female $\sqcap$ Human
$\exists r\, C$	$\{x \mid \exists y.r^I(x,y) \wedge C^I(y)\}$	existential role restriction	$\exists$married Female
$\forall r\, C$	$\{x \mid \forall y.r^I(x,y) \rightarrow C^I(y)\}$	universal role restriction	$\forall$ married Female
$A \sqsubseteq B$	$A^I \subseteq B^I$	B subsumes A	Doctor $\sqsubseteq$ Human
$A \doteq B$	$A^I \sqsubseteq B^I$ *and* $B^I \sqsubseteq A^I$	A defined by B	Parent $\doteq$ Human $\sqcap$ $\exists$hasChild Human

Fig. 8. Description logic ALC

Moreover, in Figure 9 we present a simple reasoning example. Here we are interested to check whether the concept ($\exists$married Human) subsumes the concept HappyMan which is defined in the displayed TBox as HappyMan $\doteq$ Human $\sqcap$ $\sim$Female $\sqcap$ ($\exists married$ Doctor) $\sqcap$ ($\forall$hasChild(Doctor $\sqcup$ Professor))

In Figure 10 we exemplarily prove the soundness of the standard ALC tableau rules, and we also show the correspondence between ALC and the propositional modal logic K. Note that in the embedding of propositional modal logic we here work with generic box and diamond operators which receive as first argument their accessibility relation r (of course, we could have done this also in the previous sections of these lecture notes). Apparently, from the perspective of the embeddings approach, the correspondence between ALC and propositional modal logic K becomes entirely trivial, essentially just a syntax variation. And this is exactly what the relationship between the two logics actually is.

5.2 Expressive Ontologies and Context

The study of notions of context has a long history in philosophy, linguistics, and artificial intelligence. In artificial intelligence, a major motivation has been to resolve the problem of generality of computer programs as identified by McCarthy [51]. The generality aspect of context scrutinizes flexible combinations (nestings) of contexts in combination with rich context descriptions. Giunchiglia [40] additionally emphasizes the locality aspect and the need for structured representations of knowledge. The locality aspect is particularly important for large knowledge bases, where the challenge is to effectively identify and access information that is relevant within a given reasoning context.

Different approaches to formalizing and mechanizing context have been proposed in the last decades. Many of these are outlined in the literature [3, 67, 1]. McCarthy [52] has pioneered the modeling of contexts as first class objects (in first-order logic)

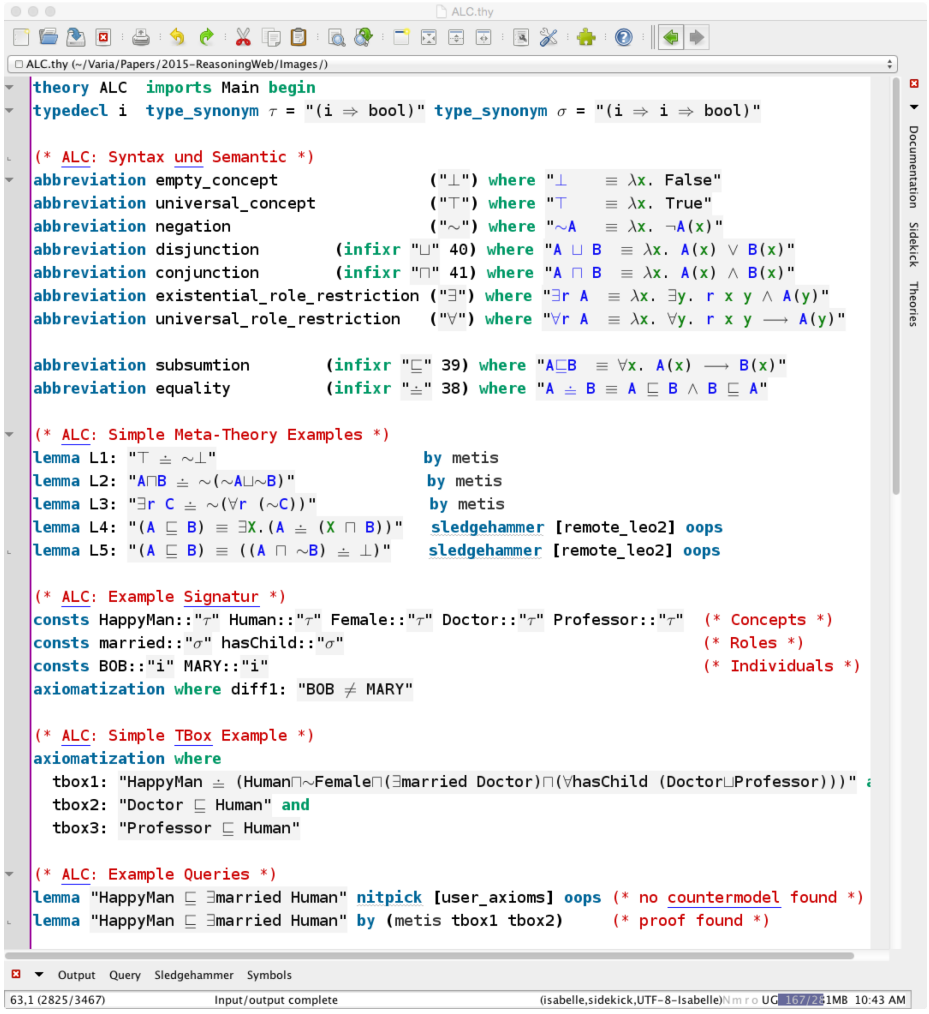


Fig. 9. Embedding of ALC in HOL

and he introduced the predicate *ist*. For example, in his approach the expression *ist(context_of("Ben's Knowledge"),likes(Sue,Bill))* encodes that proposition Sue likes Bill is true in the context of Ben's knowledge. A motivation for McCarthy's approach is actually to avoid modal logics (here for the modeling of Ben's knowledge). His line of research has been followed by a number of researchers, including, for example, Guha (who has put contexts into Cyc), Buvac and Mason [31, 44]. Also Giunchiglia and Serafini [41] avoid modal logics and propose the use of so called multilanguage systems. They show various equivalence results to common modal logics, but they also discuss several properties of multilanguage systems not supported in modal logics.

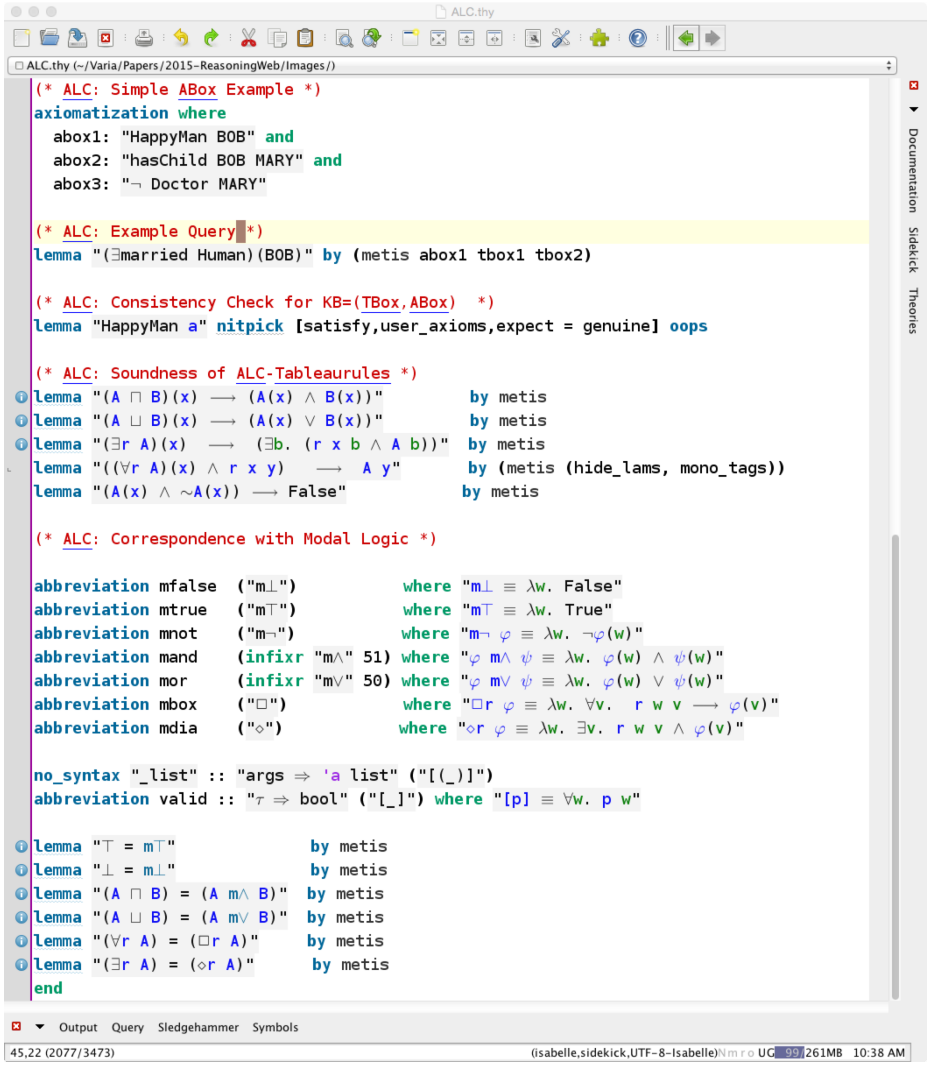


Fig. 10. Various meta-results on ALC in HOL

All of the above approaches avoid a higher-order perspective on context. However, we think that a solid higher-order perspective on context can be very valuable for various reasons. On the theory side the twist between formalisms based on modal logic and formalisms based on first-order logic seems to dissolve, since both modal logics (and other non-classical logics) and first-order logics are just natural fragments of HOL. Moreover, modal (and other) contexts can be elegantly combined and nested in HOL, so that a flexible solution to McCarthy's generality problem appears in reach. Also the

locality aspect can be addressed. The means for this is provided by relevance filtering and premise selection [4, 49].

Expressive ontologies such as the Suggested Upper Merged Ontology SUMO [59] or CYC [63] already contain a small but significant number of higher-order representations, cf. [22]. Most importantly, they employ embedded formulas (formulas at term positions), and these constructs are in fact used for modeling contexts as proposed by McCarthy, including temporal, epistemic, or doxastic contexts. The basic idea for modeling such contexts, for example, in SUMO is simple. A statement like *(loves Bill Mary)* is restricted, for instance, to the year 2009 by wrapping it (at subterm level) into respective context information:

(holdsDuring (YearFn 2009) (loves Bill Mary))

Similarly, the statement can be put into an epistemic or doxastic context:

(knows/believes Ben (loves Bill Mary))

Moreover, contexts can be flexibly combined and the embedded formulas may be complex:

(believes Bill (knows Ben (forall (?X) ((woman ?X) => (loves Bill ?X)))))

The similarity to McCarthy's approach is obvious.

Another higher-order construct used in SUMO is the set (or class) constructor *KappaFn*. It takes two arguments, a variable and a formula, and returns the set (or class) of things that satisfy the formula. Moreover, SUMO allows the use of relation and function variables.

A crucial requirement in the context of SUMO and similar expressive ontologies thus is to support flexible context reasoning in combination with other first-order and even higher-order reasoning aspects. A particular challenge thereby is to appropriately handle modal contexts, since their naive treatment may easily lead to incorrect respectively unintuitive reasoning results. As a solution we propose to encode SUMO axioms as axioms in HOML and to apply the embeddings approach to automate reasoning for SUMO. We will outline this proposal in the remainder of this section.

To illustrate the reasoning with modal contexts in SUMO we consider an example. In this example we want to answer query (C1) from axiom (A1):

(holdsDuring (YearFn 2009) (and (likes Mary Bill) (likes Sue Bill))) (A1)

(holdsDuring (YearFn 2009) (likes ?X Bill)) (C1)

The challenge is to reason about the embedded formulas within the temporal context (*holdsDuring* (*YearFn 2009*) ...). In our example, the embedded formula in the query does not match the embedded formula in the premise, however, it is inferable from it. The first-order quoting technique for reasoning with such embedded formulas presented by Pease and Sutcliffe [60], which encodes embedded formulas as strings, fails for this query. There are possible further "tricks" though which could eventually be applied. For example, we could split axiom (A1) in a pre-processing step into *(holdsDuring*

(YearFn 2009) (likes Mary Bill)) and *(holdsDuring (YearFn 2009) (likes Sue Bill))*. However, such simple tricks quickly reach their limits when considering more involved embedded reasoning problems. The following modification of our example illustrates the challenge:

(holdsDuring ?Y (likes Mary Bill)) (A2)

(holdsDuring (YearFn 2009) (forall (?X) (=> (likes Mary ?X) (likes Sue ?X)))) (A3)

(holdsDuring (YearFn ?Y) (likes Sue ?X)) (C2)

The embedded quantified formula in this example well illustrates that the reasoning tasks may quickly become non-trivial for approaches based on translations to first-order logic.

In the above examples we have (silently) assumed that the semantics of the logic underlying SUMO is a classical, bivalent logic, meaning that Boolean extensionality (BE) is valid:

$$(<=> \ (<=> ?P \ ?Q) \ (\textit{equal} \ ?P \ ?Q)) \tag{BE}$$

The left to right direction of (BE) says that there are not more than two truth values, respectively that whenever two formulas A and B can be shown equivalent then their denotations must be the same, namely either *true* or *false*. Once we have established equivalence between formulas A and B in a bivalent logic, then, in any formula C in this logic, we may substitute occurrences of A by B (and vice versa). The important aspect is that this principle not only applies to occurrences of A or B at formula level but also to occurrences at term level. For example, $(\textit{and} \ (\textit{likes Mary Bill}) \ (\textit{likes Sue Bill}))$ and $(\textit{and} \ (\textit{likes Sue Bill}) \ (\textit{likes Mary Bill}))$ are obviously equivalent, and hence, by Boolean extensionality, they have identical denotations. Thus, they can always be substituted by each other, also at the term level positions as in this situation:

(holdsDuring (YearFn 2009) (and (likes Mary Bill) (likes Sue Bill))) (A4)

(holdsDuring (YearFn 2009) (and (likes Sue Bill) (likes Mary Bill))) (C3)

Boolean extensionality seems fine for the particular temporal contexts of our previous examples. In fact, these examples have been chosen to raise the impression that Boolean extensionality is generally a natural and useful requirement for SUMO and similar ontologies. However, as we will show next, it quickly leads to counterintuitive inferences in other modal contexts. We illustrate this for epistemic and doxastic contexts. Assume that in given, concrete situation (ABox) we have:

(knows Chris (equal Chris Chris)) (A5)

(likes Mary Bill) (A6)

(knows Chris (forall (?X) (=> (likes Mary ?X) (likes Sue ?X))) (A7)

(knows Chris (likes Sue Bill)) (C4)

Assuming Boolean extensionality, the query (C4) follows from Axioms (A5)-(A7), even though we have not explicitly stated the fact $(\textit{knows Chris} \ (\textit{likes Mary Bill}))$.

Intuitively, however, assuming that Chris actually knows that Mary likes Bill seems mandatory for enabling the proof of the query. Hence, we here (re-)discover a well known issue: modalities have to be treated with great care in classical, bivalent logics.

A solution to this problem is to model SUMO's modal operators as proper modalities in HOML respectively in HOL via our embedding approach. That is, instead of translating SUMO directly into classical logic we propose to translate SUMO into HOML respectively HOL. This enables the mapping of epistemic contexts like *(knows Peter <whatever>)* or doxastic contexts like *(believes Peter <whatever>)* to proper modalities in modal logic like $\Box_{KnowledgePeter}$ *<whatever>* and $\Box_{BelievesPeter}$ *<whatever>*. The need for quantifiers and for multiple modalities is obvious from our examples so far. We may add respective axioms in order to appropriately characterize the modalities we obtain and to specify their interaction. For example, to appropriately characterize $\Box_{KnowledgePeter}$ as an epistemic modality we may use the S5 axioms and to characterize $\Box_{BelievesPeter}$ as a doxastic modality we may use the S45 axioms. Moreover, an inclusion axiom between Peter's knowledge and Peter's beliefs can be added. Alternatively, we may postulate corresponding conditions for the respective accessibility relations, cf. the symmetry condition in lines 37-14 in Figure 1 for logic KB.

We illustrate the approach with the above example. In order to capture the ABox-like status of these axioms, we introduce a fresh constant symbol cw (of world type i) to represent the current situation (as current world). The SUMO axioms (A5)-(A7) and the query (C4) are now mapped to[11]

$$(\Box_{\textit{KnowledgeChris}}\ (\textit{equal Chris Chris}))\ cw) \tag{A5}$$
$$((\textit{likes Mary Bill})\ cw) \tag{A6}$$
$$((\Box_{\textit{KnowledgeChris}})\ \textit{(forall (?X) (=> (likes Mary ?X) (likes Sue ?X)))}\ cw) \tag{A7}$$
$$((\Box_{\textit{KnowledgeChris}})\ \textit{(likes Sue Bill)}\ cw) \tag{C4}$$

Moreover, appropriate axioms need to be generated and added for each epistemic and doxastic modal operator. For example, for the epistemic modality $\Box_{\textit{KnowsChris}}$ the following S5 axioms can be added. Since these axioms are supposed to be valid in all situations (TBox-like information), they are stated with the validity operator $[.]$.

$$[\forall \phi_{\mu\to o}.\ \Box_{\textit{KnowledgeChris}}\ \phi\ \supset\ \phi]$$
$$[\forall \phi_{\mu\to o}.\ \Diamond_{\textit{KnowledgeChris}}\ \phi\ \supset\ \Box_{\textit{KnowledgeChris}}\ \Diamond_{\textit{KnowledgeChris}}\ \phi]$$

Alternatively, we may simply postulate reflexivity and seriality for the accessibility relation *KnowledgeChris*.

Subsequently the above problem can be expanded in the embeddings approach into a proper HOL encoding, and then HOL reasoners can be applied for proving or refuting it. In fact, the mapped example in HOML is not valid and HOL-ATPs are able to detect a counter model, which is what we wanted to achieve. However, if we replace (A6) by $(\Box_{\textit{KnowledgeChris}}$ *(likes Mary Bill) cw)*, then the problem can be quickly proved.

[11] More elegantly, we could employ an $@_{cw}$-operator; for example, (A6) would then be encoded as $@_{cw}(likesMaryBill)$ (see also Section 5.4).

Note that the sketched approach scales for other modal operators in SUMO besides *knows* and *believes*. Most importantly, it even supports their flexible combination and bridge rules can be easily postulated.

5.3 Metaphysics

In this subsection we illustrate the use of the embeddings approach for the formalization and verification of Scott's version [66] of Gödel's ontological argument for God's existence [13, 19]. This proof was chosen mainly for two reasons. Firstly, it requires not only modal operators, but also higher-order quantification. Therefore, it is beyond the reach of specialized propositional and first-order (modal) theorem provers. Secondly, this argument addresses an ancient problem in Philosophy and Metaphysics, which has nevertheless received a lot of attention in the last 15 years, because of the discovery of the modal collapse [69, 70]. This proof lies in the center of a vast and largely unexplored application domain for automated and interactive theorem provers.

Attempts to prove the existence (or non-existence) of God by means of abstract ontological arguments are an old tradition in philosophy and theology. Gödel's proof [42] is a modern culmination of this tradition, following particularly the footsteps of Leibniz. Various slightly different versions of axioms and definitions have been considered by Gödel and by several philosophers who commented on his proof (cf. [70, 5, 36, 2, 33]).

Thanks to the embedding approach, Gödel's theorem stating God's necessary existence was automatically proven from his five axioms using fully automated higher-order theorem provers [13, 19].

The respective encodings and the results of a series of recent experiments with LEO-II (version 1.6.2), Satallax (version 2.7), and Nitpick (version 2013) are provided in Fig. 12. The first row marked with T1, for example, shows that theorem T1 follows from axioms A2 and A1 (where only the $\supset$-direction is needed); LEO-II and Satallax confirm this in 0.1 second. The experiments have been carried out w.r.t. the logics K and/or KB, and w.r.t. constant (const) and varying (vary) domain semantics for the domains of individuals. The exact dependencies (available axioms and definitions) are displayed for each single problem. The results of the prover calls are given in seconds. '—' means timeout. 'THM', 'CSA', 'SAT', and 'UNS' are the reported result statuses; they stand for 'Theorem', 'CounterSatisfiable', 'Satisfiable', and 'Unsatisfiable', respectively. The experiments were executed remotely using calls to LEO-II, Satallax, and Nitpick installed at Sutcliffe's SystemOnTPTP infrastructure [71] at the University of Miami, which comprises of standard 2.80GHz computers with 1GB memory. An example problem from these experiments has been presented in Figure 2.

Several interesting and partly novel findings have been discovered by the HOL-ATPs, including:

1. The axioms and definitions from Fig. 11 are consistent (cf. CO in Fig. 12).
2. Logic K is sufficient for proving T1, C and T2.
3. For proving the final theorem T3, logic KB is sufficient (and for K a countermodel is reported). This is highly relevant since several philosophers have criticized Gödel's argument for the use of logic S5.

A1 Either a property or its negation is positive, but not both:

$$\forall \phi[\boldsymbol{P}(\neg\phi) \equiv \neg \boldsymbol{P}(\phi)]$$

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

$$\forall \phi \forall \psi[(\boldsymbol{P}(\phi) \wedge \Box \forall \boldsymbol{x}[\phi(\boldsymbol{x}) \supset \psi(\boldsymbol{x})]) \supset \boldsymbol{P}(\psi)]$$

T1 Positive properties are possibly exemplified:

$$\forall \phi[\boldsymbol{P}(\phi) \supset \Diamond \exists \boldsymbol{x} \phi(\boldsymbol{x})]$$

D1 A *God-like* being possesses all positive properties:

$$\boldsymbol{G}(\boldsymbol{x}) \equiv \forall \phi[\boldsymbol{P}(\phi) \supset \phi(\boldsymbol{x})]$$

A3 The property of being God-like is positive:

$$\boldsymbol{P}(\boldsymbol{G})$$

C Possibly, God exists:

$$\Diamond \exists \boldsymbol{x} \boldsymbol{G}(\boldsymbol{x})$$

A4 Positive properties are necessarily positive:

$$\forall \phi[\boldsymbol{P}(\phi) \supset \Box\, \boldsymbol{P}(\phi)]$$

D2 An *essence* of an individual is a property possessed by it and necessarily implying any of its properties:

$$\phi\ \boldsymbol{ess.}\ \boldsymbol{x} \equiv \phi(\boldsymbol{x}) \wedge \forall \psi(\psi(\boldsymbol{x}) \supset \Box \forall \boldsymbol{y}(\phi(\boldsymbol{y}) \supset \psi(\boldsymbol{y})))$$

T2 Being God-like is an essence of any God-like being:

$$\forall \boldsymbol{x}[\boldsymbol{G}(\boldsymbol{x}) \supset \boldsymbol{G}\ \boldsymbol{ess.}\ \boldsymbol{x}]$$

D3 *Necessary existence* of an individ. is the necessary exemplification of all its essences:

$$\boldsymbol{NE}(\boldsymbol{x}) \equiv \forall \phi[\phi\ \boldsymbol{ess.}\ \boldsymbol{x} \supset \Box \exists \boldsymbol{y} \phi(\boldsymbol{y})]$$

A5 Necessary existence is a positive property:

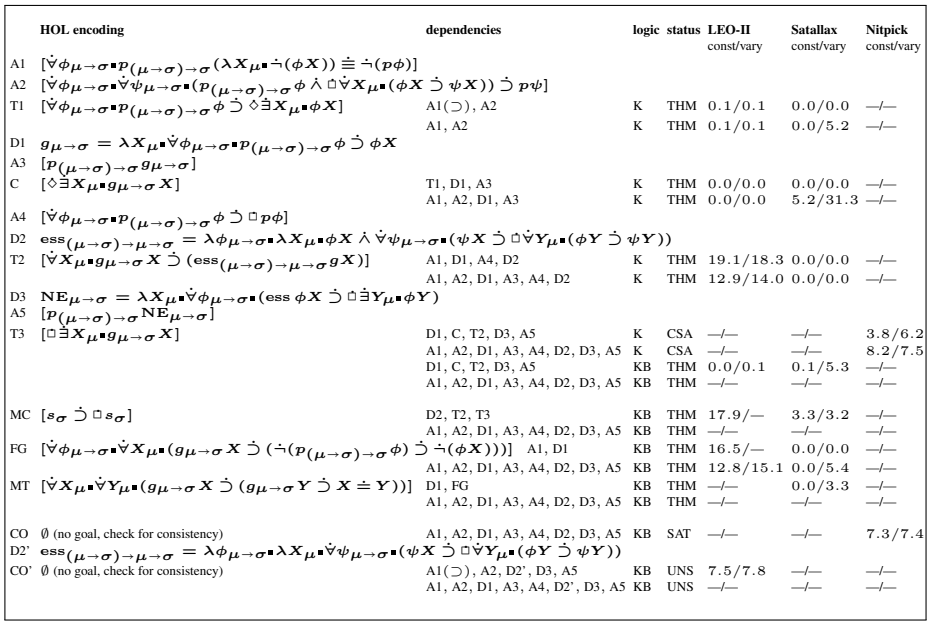
$$\boldsymbol{P}(\boldsymbol{NE})$$

T3 Necessarily, God exists:

$$\Box \exists \boldsymbol{x} \boldsymbol{G}(\boldsymbol{x})$$

Fig. 11. Scott's version of Gödel's ontological argument [66].

4. Only for T3 the HOL-ATPs still fail to produce a proof directly from the axioms; thus, T3 remains an interesting benchmark problem; T1, C, and T2 are rather trivial for HOL-ATPs.



	HOL encoding	dependencies	logic	status	LEO-II const/vary	Satallax const/vary	Nitpick const/vary
A1	$[\dot{\forall}\phi_{\mu\to\sigma}.p_{(\mu\to\sigma)\to\sigma}(\lambda X_\mu.\dot{\neg}(\phi X)) \doteq \dot{\neg}(p\phi)]$						
A2	$[\dot{\forall}\phi_{\mu\to\sigma}.\dot{\forall}\psi_{\mu\to\sigma}.(p_{(\mu\to\sigma)\to\sigma}\phi \dot{\wedge} \Box\dot{\forall}X_\mu.(\phi X \dot{\supset} \psi X)) \dot{\supset} p\psi]$						
T1	$[\dot{\forall}\phi_{\mu\to\sigma}.p_{(\mu\to\sigma)\to\sigma}\phi \dot{\supset} \Diamond\dot{\exists}X_\mu.\phi X]$	A1($\supset$), A2	K	THM	0.1/0.1	0.0/0.0	—/—
		A1, A2	K	THM	0.1/0.1	0.0/5.2	—/—
D1	$g_{\mu\to\sigma} = \lambda X_\mu.\dot{\forall}\phi_{\mu\to\sigma}.p_{(\mu\to\sigma)\to\sigma}\phi \dot{\supset} \phi X$						
A3	$[p_{(\mu\to\sigma)\to\sigma}g_{\mu\to\sigma}]$						
C	$[\Diamond\dot{\exists}X_\mu.g_{\mu\to\sigma}X]$	T1, D1, A3	K	THM	0.0/0.0	0.0/0.0	—/—
		A1, A2, D1, A3	K	THM	0.0/0.0	5.2/31.3	—/—
A4	$[\dot{\forall}\phi_{\mu\to\sigma}.p_{(\mu\to\sigma)\to\sigma}\phi \dot{\supset} \Box p\phi]$						
D2	$\mathbf{ess}_{(\mu\to\sigma)\to\mu\to\sigma} = \lambda\phi_{\mu\to\sigma}.\lambda X_\mu.\phi X \dot{\wedge} \dot{\forall}\psi_{\mu\to\sigma}.(\psi X \dot{\supset} \Box\dot{\forall}Y_\mu.(\phi Y \dot{\supset} \psi Y))$						
T2	$[\dot{\forall}X_\mu.g_{\mu\to\sigma}X \dot{\supset} (\mathbf{ess}_{(\mu\to\sigma)\to\mu\to\sigma}gX)]$	A1, D1, A4, D2	K	THM	19.1/18.3	0.0/0.0	—/—
		A1, A2, D1, A3, A4, D2	K	THM	12.9/14.0	0.0/0.0	—/—
D3	$\mathbf{NE}_{\mu\to\sigma} = \lambda X_\mu.\dot{\forall}\phi_{\mu\to\sigma}.(\mathbf{ess}\,\phi X \dot{\supset} \Box\dot{\exists}Y_\mu.\phi Y)$						
A5	$[p_{(\mu\to\sigma)\to\sigma}\mathbf{NE}_{\mu\to\sigma}]$						
T3	$[\Box\dot{\exists}X_\mu.g_{\mu\to\sigma}X]$	D1, C, T2, D3, A5	K	CSA	—/—	—/—	3.8/6.2
		A1, A2, D1, A3, A4, D2, D3, A5	K	CSA	—/—	—/—	8.2/7.5
		D1, C, T2, D3, A5	KB	THM	0.0/0.1	0.1/5.3	—/—
		A1, A2, D1, A3, A4, D2, D3, A5	KB	THM	—/—	—/—	—/—
MC	$[s_\sigma \dot{\supset} \Box s_\sigma]$	D2, T2, T3	KB	THM	17.9/—	3.3/3.2	—/—
		A1, A2, D1, A3, A4, D2, D3, A5	KB	THM	—/—	—/—	—/—
FG	$[\dot{\forall}\phi_{\mu\to\sigma}.\dot{\forall}X_\mu.(g_{\mu\to\sigma}X \dot{\supset} (\dot{\neg}(p_{(\mu\to\sigma)\to\sigma}\phi) \dot{\supset} \dot{\neg}(\phi X)))]$	A1, D1	KB	THM	16.5/—	0.0/0.0	—/—
		A1, A2, D1, A3, A4, D2, D3, A5	KB	THM	12.8/15.1	0.0/5.4	—/—
MT	$[\dot{\forall}X_\mu.\dot{\forall}Y_\mu.(g_{\mu\to\sigma}X \dot{\supset} (g_{\mu\to\sigma}Y \dot{\supset} X \doteq Y))]$	D1, FG	KB	THM	—/—	0.0/3.3	—/—
		A1, A2, D1, A3, A4, D2, D3, A5	KB	THM	—/—	—/—	—/—
CO	$\emptyset$ (no goal, check for consistency)	A1, A2, D1, A3, A4, D2, D3, A5	KB	SAT	—/—	—/—	7.3/7.4
D2'	$\mathbf{ess}_{(\mu\to\sigma)\to\mu\to\sigma} = \lambda\phi_{\mu\to\sigma}.\lambda X_\mu.\dot{\forall}\psi_{\mu\to\sigma}.(\psi X \dot{\supset} \Box\dot{\forall}Y_\mu.(\phi Y \dot{\supset} \psi Y))$						
CO'	$\emptyset$ (no goal, check for consistency)	A1($\supset$), A2, D2', D3, A5	KB	UNS	7.5/7.8	—/—	—/—
		A1, A2, D1, A3, A4, D2', D3, A5	KB	UNS	—/—	—/—	—/—

Fig. 12. HOL encodings and experiment results for the ontological argument from Fig. 11.

5. Gödel's original version of the proof [43], which omits conjunct $\phi(x)$ in the definition of *essence* (cf. D2'), seems inconsistent (cf. the failed consistency check for CO' in Fig. 12). As far as we are aware of, this is a new result.
6. Gödel's axioms imply what is called the modal collapse (cf. MC in Fig. 12) $\phi \supset \Box\phi$, that is, contingent truth implies necessary truth (which can even be interpreted as an argument against free will; cf. [70]). MC is probably the most fundamental criticism put forward against Gödel's argument.
7. All of the above findings hold for both constant domain semantics and varying domain semantics (for the domain of individuals).

The above findings, in particular (7), illustrate that the modal reasoning framework described here has a great potential towards a flexible support system for *computational theoretical philosophy*. In fact, Gödel's ontological argument has been verified and even automated not only for one particular setting of logic parameters, but these logic parameters have been varied and the validity of the argument has been reconfirmed (or falsified, cf. D2' and CO') for the modified setting. Moreover, our framework is not restricted to a particular theorem proving system, but has been fruitfully employed with some of the most prominent automated theorem provers available to date. A semi-automatic verification of Gödel's argument was also realized in `Isabelle`, with partial automation via Sledgehammer, Nitpick and Metis (see Figures 3 and 4) [18].

When a fully automatic or semi-automatic verification is performed, the formal proof structure is hidden and may not correspond to the informal structure of the ar-

gument. In order to verify the exact argument in all detail, a fully interactive and fine-grained formalization is needed. We show and discuss such a formalization in Coq (version 8.4pl5) below. In contrast to the formalization in Isabelle [18], the formalization in Coq used no automation. This was a deliberate choice, mainly because it allowed a qualitative evaluation of the convenience of the embedding approach for *interactive* theorem proving.

The formalization shown below aims at being as similar as possible to Dana Scott's version of the proof [66]. The formulation and numbering of axioms, definitions and theorems is the same as in Scott's notes. Even the Coq proof scripts follow precisely all the steps in Scott's notes. Scott's assertions are emphasized below with comments. Furthermore, the deliberate preference for simple tactics (mostly *intro*, *apply* and the modal tactics described in Section 4.3) results in proof scripts that closely correspond to common natural deduction proofs.

Gödel's proof requires Coq's classical logic libraries as well as the Modal library developed by us and described in Section 4.3.

```
Require Import Coq.Logic.Classical Coq.Logic.Classical_Pred_Type Modal.
```

In Scott's notes, classicality occurs in uses of the principle of proof by contradiction. In order to clearly indicate where classical logic is needed in the proof scripts, a simple tactic that simulates proof by contradiction was created:

```
Ltac proof_by_contradiction H := apply NNPP; intro H.
```

Gödel's theory has a single higher-order constant, Positive, which ought to hold for properties considered *positive* in a moral sense.

```
(* Constant predicate that distinguishes positive properties *)
Parameter Positive: (u -> o) -> o.
```

God is defined as a being possessing all positive properties, and five axioms are stated to characterize positivity. The first part of the proof culminates in corollary1 and establishes that God's existence is possible.

```
(* Axiom A1 (divided into two directions):
   either a property or its negation is positive, but not both *)
Axiom axiom1a :
  [ mforall p, (Positive (fun x: u => m~(p x))) m-> (m~ (Positive p)) ].

Axiom axiom1b :
  [ mforall p, (m~ (Positive p)) m-> (Positive (fun x: u => m~ (p x))) ].

(* Axiom A2:
   a property necessarily implied by a positive property is positive *)
Axiom axiom2: [ mforall p, mforall q,
  Positive p m/\ (box (mforall x, (p x) m-> (q x) )) m-> Positive q ].

(* Theorem T1: positive properties are possibly exemplified *)
Theorem theorem1: [ mforall p, (Positive p) m-> dia (mexists x, p x) ].
Proof. mv.
intro p. intro H1. proof_by_contradiction H2. apply not_dia_box_not in H2.
assert (H3: ((box (mforall x, m~ (p x))) w)). (* Scott *)
  box_i. intro x. assert (H4: ((m~ (mexists x : u, p x)) w0)).
    box_e H2 G2. exact G2.
    clear H2 R H1 w. intro H5. apply H4. exists x. exact H5.
  assert (H6: ((box (mforall x, (p x) m-> m~ (x m= x))) w)). (* Scott *)
```

```
    box_i. intro x. intros H7 H8. box_elim H3 w0 G3. eapply G3. exact H7.
    assert (H9: ((Positive (fun x => m~ (x m= x))) w)). (* Scott *)
      apply (axiom2 w p (fun x => m~ (x m= x))). split.
        exact H1.
        exact H6.
      assert (H10: ((box (mforall x, (p x) m-> (x m= x))) w)). (* Scott *)
        box_i. intros x H11. reflexivity.
        assert (H11 : ((Positive (fun x => (x m= x))) w)). (* Scott *)
          apply (axiom2 w p (fun x => x m= x )). split.
            exact H1.
            exact H10.
          apply axiom1a in H9. contradiction.
Qed.

(* Definition D1:
   God: a God-like being possesses all positive properties *)
Definition G(x: u) := mforall p, (Positive p) m-> (p x).

(* Axiom A3: the property of being God-like is positive *)
Axiom axiom3: [ Positive G ].

(* Corollary C1: possibly, God exists *)
Theorem corollary1: [ dia (mexists x, G x) ].
Proof. mv. apply theorem1. apply axiom3. Qed.
```

The second part of the proof consists in showing that if God's existence is possible then it must be necessary (`lemma2`). The controversial **S5** principle `dia_box_to_box` is used.

```
(* Axiom A4: positive properties are necessarily positive *)
Axiom axiom4: [ mforall p, (Positive p) m-> box (Positive p) ].

(* Definition D2:
   essence: an essence of an individual is a property possessed by it
   and necessarily implying any of its properties *)
Definition Essence(p: u -> o)(x: u) :=
    (p x) m/\ mforall q, ((q x) m-> box (mforall y, (p y) m-> (q y))).
Notation "p 'ess' x" := (Essence p x) (at level 69).

(* Theorem T2: being God-like is an essence of any God-like being *)
Theorem theorem2: [ mforall x, (G x) m-> (G ess x) ].
Proof. mv. intro g. intro H1. unfold Essence. split.
  exact H1.
  intro q. intro H2. assert (H3: ((Positive q) w)).
    proof_by_contradiction H4. unfold G in H1. apply axiom1b in H4.
    apply H1 in H4. contradiction.

    cut (box (Positive q) w). (* Scott *)
      apply K. box_i. intro H5. intro y. intro H6.
      unfold G in H6. apply (H6 q). exact H5.

      apply axiom4. exact H3.
Qed.

(* Definition D3:
   necessary existence: necessary existence of an individual
   is the necessary exemplification of all its essences *)
Definition NE(x: u) := mforall p, (p ess x) m-> box (mexists y, (p y)).

(* Axiom A5: necessary existence is a positive property *)
Axiom axiom5: [ Positive NE ].

Lemma lemma1: [ (mexists z, (G z)) m-> box (mexists x, (G x)) ].
Proof. mv.
intro H1. destruct H1 as [g H2]. cut ((G ess g) w). (* Scott *)
  assert (H3: (NE g w)).       (* Scott *)
```

```
    unfold G in H2. apply (H2 NE). apply axiom5.
    unfold NE in H3. apply H3.
  apply theorem2. exact H2.
Qed.

Lemma lemma2: [ dia (mexists z, (G z)) m-> box (mexists x, (G x)) ].
Proof. mv.
intro H. cut (dia (box (mexists x, G x)) w).  (* Scott *)
  apply dia_box_to_box.
  apply (mp_dia w (mexists z, G z)).
    exact H.
    box_i. apply lemma1.
Qed.

(* Theorem T3: necessarily, a God exists *)
Theorem theorem3: [ box (mexists x, (G x)) ].
Proof. mv. apply lemma2. apply corollary1. Qed.

(* Corollary C2: There exists a god *)
Theorem corollary2: [ mexists x, (G x) ].
Proof. mv. apply T. apply theorem3. Qed.
```

5.4 Paraconsistent Reasoning through Higher-Order Hybrid Logics

Inconsistencies pose a significant challenge to proper reasoning in the web. It is well-known that classical logic validates the *principle of explosion*, according to which every proposition follows from a contradiction (*ex contractione quodlibet*). Hence, any inconsistency in the vast knowledge available in the web, no matter how tiny, insignificant, unreliable, exceptional or irrelevant it is to our query, would render classical reasoners useless.

Several approaches have been proposed to overcome this challenge. The diversity of approaches reflects the large variety of kinds of inconsistency that we can encounter. For example, contradictory pieces of information may be due to errors made by ourselves; or they may merely express divergent opinions from other sources. Or perhaps two statements may contradict one another because one of them expresses a general rule that is not always applicable, while the other describes an exception to the rule. If we adhered to dialetheism, contradictions in a theory could even be taken to reflect actual contradictions in models where statement could be simultaneously true and false. Depending on the situation, we may wish to, for instance, *revise* the data (i.e. our *beliefs*) [61, 65], do *default reasoning* preferring exceptions to general rules, simply ignore contradictions when they are *irrelevant* [62, 35] to the reasoning task at hand, or use non-classical *paraconsistent logics* that block the principle of explosion [25, 34, 73].

In this section we informally sketch a logic that is adequate for applications where data originates from different independent sources, which are assumed to be separately consistent but possibly mutually inconsistent. Such a scenario is common in the web, where we must do the best reasoning we can despite the limited control over the information provided by (often not fully trusted) data sources. The basic idea of this logic goes back to the modal *discussive logics* of Jaskowski [46, 47], in which the fact that a participant/source claims p is expressed by $\Diamond p$. These logics exhibit a paraconsistent behavior in the following sense: if two participants make contradictory claims such as q and $\neg q$, an arbitrary proposition r is not implied, because $\Diamond q \wedge \Diamond \neg q \supset r$ is not

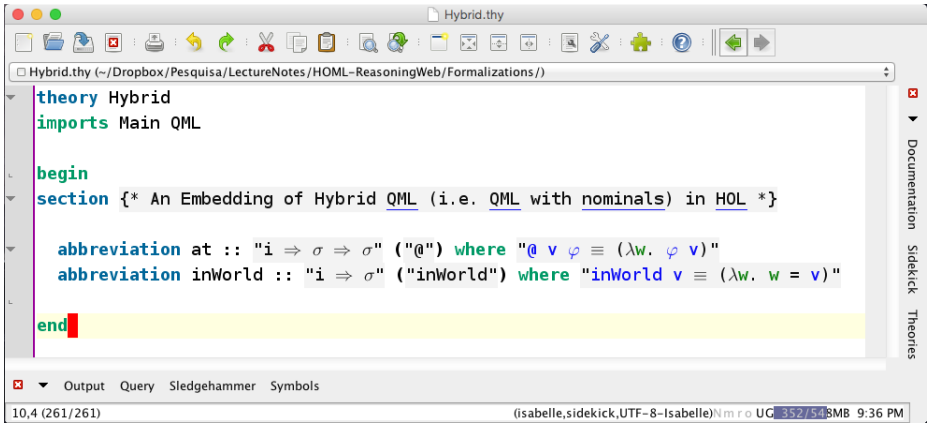


Fig. 13. Higher-Order Hybrid Logics

valid. Jaskowski's logics assume the modal axiom T ($\Box p \supset p$), which in this context expresses the fact that a proposition holds if all participants unanimously claim it.

In Jaskowski's discussive logics each participant/source is a distinct possible world. Their main limitation is the impossibility of referring to each participant/source/world explicitly, because $\Diamond$ and $\Box$ are the only available modalities. A problematic consequence of this parsimony is, for instance, the lack of modus ponens relative to the claims of a single participant: if a participant claims p and later claims $p \supset q$, these claims are formalized as $\Diamond p$ and $\Diamond(p \supset q)$; but then, unfortunately, $\Diamond q$ cannot be deduced. Discussive logics try to remedy this problem in ways (cf. [10]) that may seem unnatural and unnecessary after the advent of hybrid logics, which extend modal logics with nominals and the @ modality. In a hybrid discussive logic, as proposed here, the claims p and $p \supset q$ by a participant j could be formalized as $@_j p$ and $@_j(p \supset q)$. Hence, information about who claimed what is preserved. Furthermore, the @ modality gives greater flexibility and control over which sources/participants to trust. In addition to trusting the consensus ($\Box p \supset p$), it becomes possible to declare that a particular source s is trusted, by stating that ($@_s p \supset p$).

With the embedding approach, it is trivial to define nominals and the @ modality, because worlds are already syntactically explicit. This is shown in Figure 13.

Note that we have in fact already employed the @ modality in Section 5.2 in the axioms (A5)-(A7) and conjecture (C4).

Once we have a higher-order hybrid modal logic at our disposal, we assign each information source to a different world, and we may reason explicitly about inconsistencies between the information sources. This is shown in Figure . The fact that Nitpick finds a counter-model for the principle of explosion demonstrates that this logic is paraconsistent.

Figure 15 shows a toy example of reasoning with two sources of information, the TV channels *CNN* and *Russia Today* (*RT*), which disagree with each other about the qualities of the president of Russia. If we simply believed everything that we hear from CNN and RT, our beliefs would be inconsistent. The hybrid modal logic proposed here allow

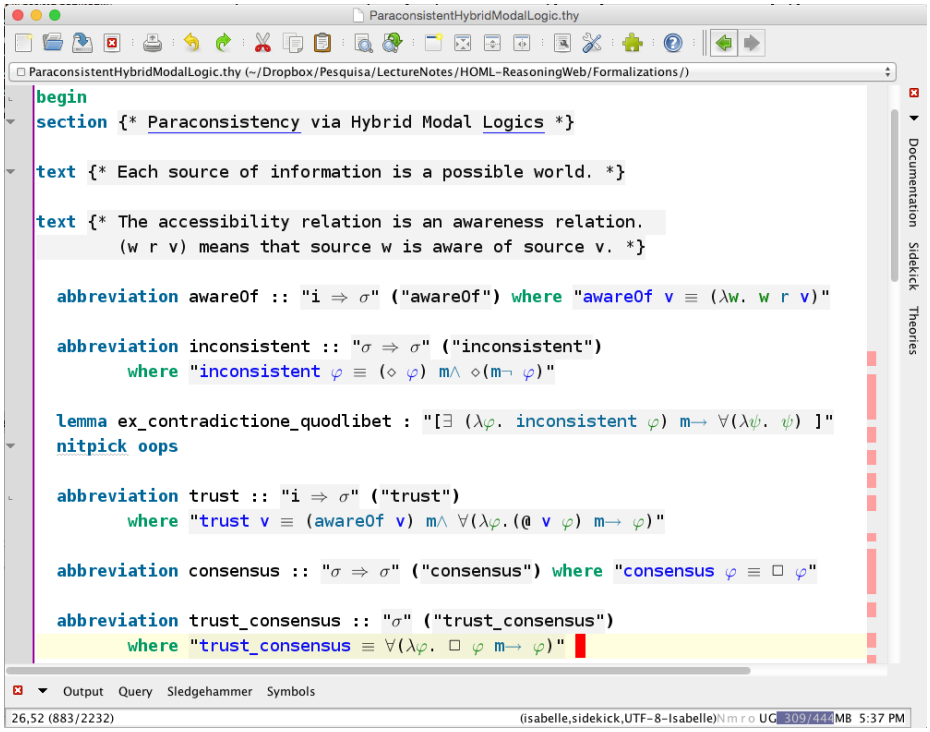


Fig. 14. Paraconsistency

us to be skeptikal about our information sources and possibly choose which source we would like to trust.

Distinct sources of information may disagree not only on the propositional level but also on their understanding of properties and individuals. Consider, for example, a model with two worlds: m (Mars) and e (Earth); and consider whether the sentence $\Box blue^{\star}(sky^{\star})$ ("necessarily the sky is blue") is true in this model. From an external absolute perspective, the sky is assumed to be $blue$ in Earth and red in Mars, and therefore, if $blue^{\star}$ were considered to be a *rigid property* (equal to the absolute notion of "blue", so that $blue^{\star} = blue$), the sentence would be clearly false, because the sky is not $blue$ in Mars. However, if $blue^{\star}$ were considered to be a *flexible property* (i.e. depending on worlds), the sentence would be ambiguous. In fact, also $sky^{\star}$ refers to a different thing in each world and hence it could also be considered as a *flexible individual* dependent on worlds. These ambiguities become clear when we try to translate the sentence to classical higher-order logic and they are resolved when we opt for one of the following four possible translations:

A: $\forall\, w\, w'.(rww') \supset (((\boldsymbol{blue}^{\star}\ w')(\boldsymbol{sky}^{\star}\ w'))\ w')$
B: $\forall\, w\, w'.(rww') \supset (((\boldsymbol{blue}^{\star}\ w)(\boldsymbol{sky}^{\star}\ w'))\ w')$
C: $\forall\, w\, w'.(rww') \supset (((\boldsymbol{blue}^{\star}\ w')(\boldsymbol{sky}^{\star}\ w))\ w')$

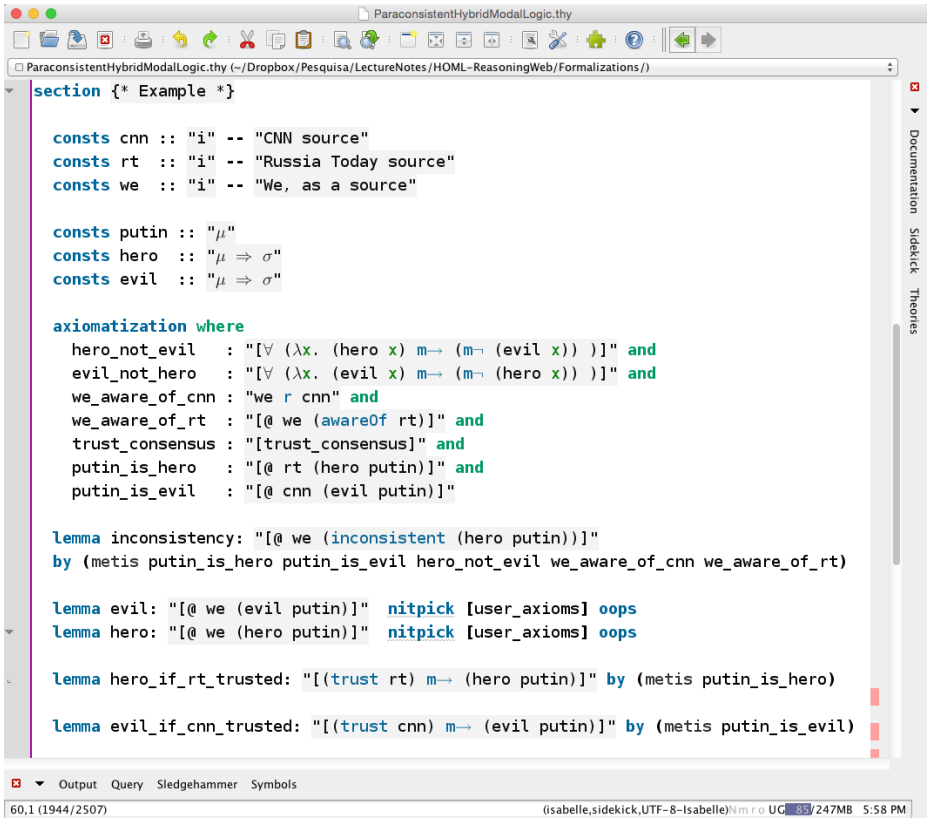


Fig. 15. An example of conflicting sources of information

D: $\forall\, w\, w'.(rww') \supset (((\boldsymbol{blue}^\star\ w)(\boldsymbol{sky}^\star\ w))\ w')$

These translations differ on the world that is taken for grounding flexible constants (e.g. $\boldsymbol{blue}^\star$ and $\boldsymbol{sky}^\star$) under the scope of modal operators. Translation A, for example, grounds the flexible constants on the world w' introduced by the modal operator, while translation D grounds them on the current world w. If the Martian understanding of $\boldsymbol{blue}^\star$ is equal to the absolute notion of red (i.e. $(\boldsymbol{blue}^\star\ m) = \boldsymbol{red}$) (and, likewise for Earth, $(\boldsymbol{blue}^\star\ e) = \boldsymbol{blue}$), then translation A would be true.

Flexible constants (properties or individuals) may be useful for reasoning with many knowledge bases (e.g. in description logics or expressive ontologies such as SUMO) having overlapping names for concepts or objects. Merely merging these knowledge bases could easily lead to inconsistencies (e.g. with the sky being both blue, according to the Earthling knowledge base, and not blue, according to the Martian knowledge base). Instead, embedding these knowledge bases into a higher-order hybrid logic, with each knowledge base occupying a separate world and flexible constants used for conflicting concepts and objects, provides a simple and safe alternative to avoid inconsistencies.

The language of quantified modal logic defined in Section 2 does not allow the user to specify on which world a flexible property or object should be grounded. There is also no way for flexible and rigid properties/objects to be used together. The semantic embedding described in Section 3.2 assumes that they are all rigid. An alternative would be to assume that they are flexible and ground them on the world introduced by the closest modal operator (e.g. as in translation A). Fitting [36] discusses yet another possibility, at least for certain kinds of properties introduced by higher-order quantifiers: they are grounded to the world where they were introduced (i.e. a modal formula such as $\exists\psi.\Box\psi(c)$ (where c is assumed to be a rigid constant) would be translated[12] as $\forall \boldsymbol{w}.\exists \boldsymbol{\psi}.\forall \boldsymbol{w'}.(\boldsymbol{r}\ \boldsymbol{w}\ \boldsymbol{w'}) \supset (\boldsymbol{\psi w})(\boldsymbol{c})$, and not as $\forall \boldsymbol{w}.\exists \boldsymbol{\psi}.\forall \boldsymbol{w'}.(\boldsymbol{r}\ \boldsymbol{w}\ \boldsymbol{w'}) \supset (\boldsymbol{\psi w'})(\boldsymbol{c})$ because w is the world having scope over the existential quantifier introducing ψ).

Instead of being content with the language of higher-order modal logic from Section 3.2 and then choosing either the rigid translation or some flexible translation, an even more interesting possibility, whose details remain for future work, would be to enrich the language of higher-order quantified modal logic, in order to empower the user to conveniently specify how flexible terms should be grounded. What currently prevents this is that the worlds implicitly introduced by modal operators are hidden. Therefore, one approach would be to enrich the language with modal operators that explicitly expose the introduced worlds. The four alternative disambiguations of $\Box blue^{\star}(sky^{\star})$ discussed above, for example, could then be written as follows in the enriched language:

A: $\Box_{w'} blue^{w'}(sky^{w'})$
B: $\Box_{w'} blue^{w}(sky^{w'})$
C: $\Box_{w'} blue^{w'}(sky^{w})$
D: $\Box_{w'} blue^{w}(sky^{w})$

where w would be the current world, by convention.

With embeddings, enriching the language in this manner is easy, as shown in Figures 16 and 17.

Another approach would be to use a nameless bound variable notation [32] for worlds, as follows:

A: $\Box blue^{0}(sky^{0})$
B: $\Box blue^{1}(sky^{0})$
C: $\Box blue^{0}(sky^{1})$
D: $\Box blue^{1}(sky^{1})$

where the superscript indices are de-Bruijn indices indicating the nameless bound world that should be used for grounding.

Achieving such nameless notation in Isabelle is made possible by the advanced feature of syntax translations, as exemplified in the formalization of Hoare Logic [74].

[12] Fitting [36](pp. 83ff) actually does not use a translation to higher-order logic, where worlds become part of the syntax. But what he does, using his style of syntax (which distinguishes extensional and intensional types), is essentially analogous to the translation described here.

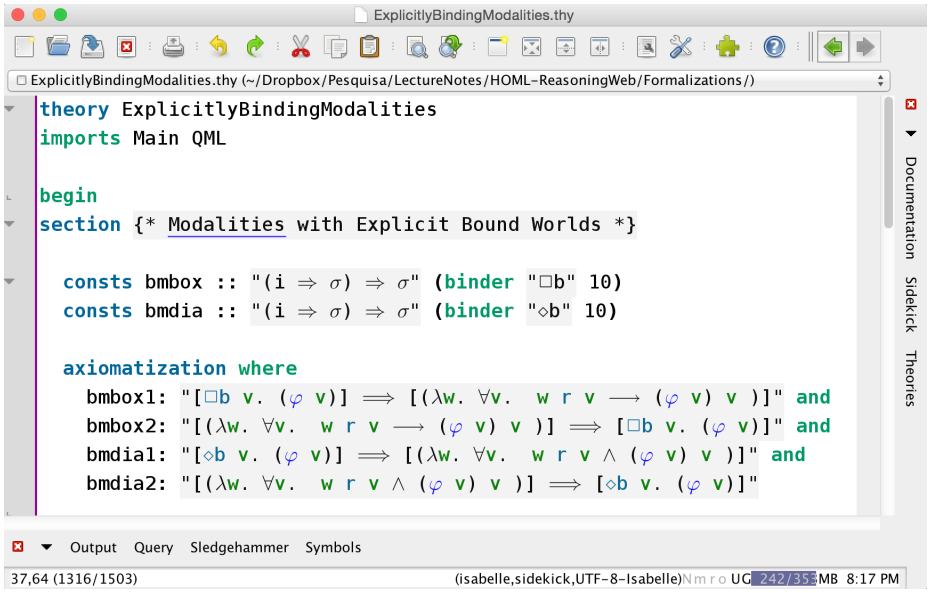


Fig. 16. Explicitly binding modalities

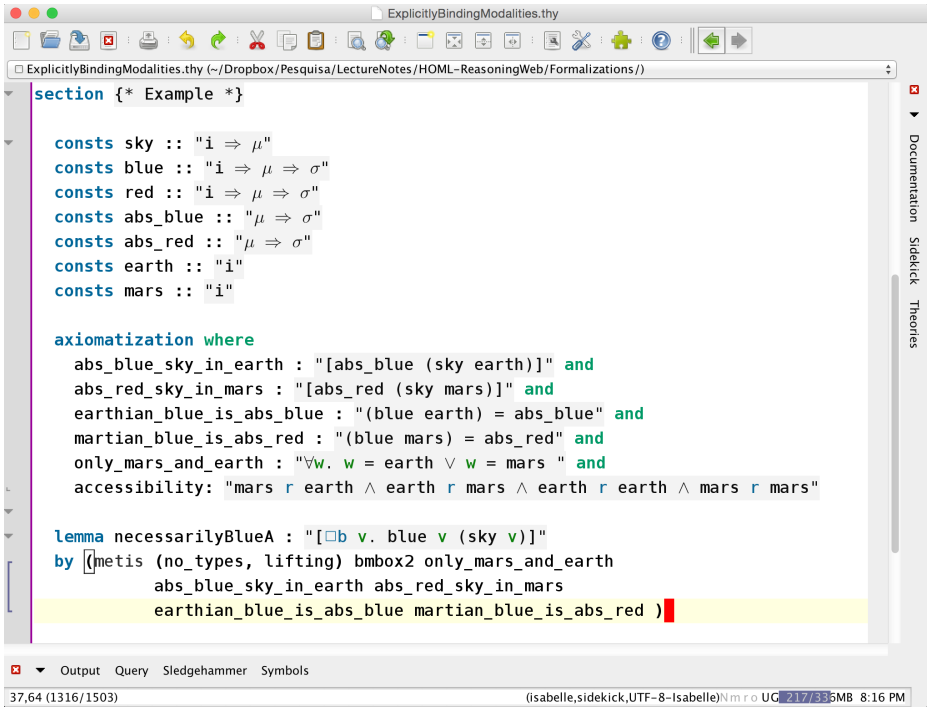


Fig. 17. Example using explicitly binding modalities

6 Conclusion

In these lecture notes, we have explained the latest developments in automated reasoning for higher-order modal logics. We have also surveyed recent and potential applications of such expressive logics. This is a vast and exciting direction of research, which has become possible by the high degree of maturity achieved by current higher-order theorem provers and proof assistants.

Acknowledgments: We would like to thank João Marcos for consistently useful discussions about discussive logics and paraconsistency. Various persons have contributed or positively influenced this line of research in the past, including, Larry Paulson, Chad Brown, Geoff Sutcliffe, and Jasmin Blanchette.

References

1. Web semantics: Science, services and agents on the world wide web, special issue on reasoning with context in the semantic web, volumes 12–13, pages 1-160, 2012.
2. R.M. Adams. Introductory note to *1970. In *Kurt Gödel: Collected Works Vol. 3: Unpubl. Essays and Letters*. Oxford Univ. Press, 1995.
3. V. Akman and M. Surav. Steps toward formalizing context. *AI Magazine*, 17(3), 1996.
4. Jesse Alama, Tom Heskes, Daniel Kühlwein, Evgeni Tsivtsivadze, and Josef Urban. Premise selection for mathematics by corpus analysis and kernel methods. *Journal of Automated Reasoning*, 52(2):191–213, 2014.
5. A.C. Anderson and M. Gettings. Gödel ontological proof revisited. In *Gödel'96: Logical Foundations of Mathematics, Computer Science, and Physics: Lecture Notes in Logic 6*, pages 167–172. Springer, 1996.
6. P.B. Andrews. General models and extensionality. *Journal of Symbolic Logic*, 37(2):395–397, 1972.
7. P.B. Andrews. Church's type theory. In E.N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Spring 2014 edition, 2014.
8. Peter B. Andrews, Dale A. Miller, Eve Longini Cohen, and Frank Pfenning. Automating higher-order logic. In W. W. Bledsoe and D. W. Loveland, editors, *Automated Theorem Proving: After 25 Years*, volume 29 of *Contemporary Mathematics series*, pages 169–192. American Mathematical Society, 1984.
9. PeterB. Andrews and Matthew Bishop. On sets, types, fixed points, and checkerboards. In P. Miglioli, U. Moscato, D. Mundici, and M. Ornaghi, editors, *Theorem Proving with Analytic Tableaux and Related Methods*, volume 1071 of *Lecture Notes in Computer Science*, pages 1–15. Springer Berlin Heidelberg, 1996.
10. Jo ao Marcos. Modality and paraconsistency. *The Logica Yearbook*, pages 213–222, 2005.
11. Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, New York, NY, USA, 2003.
12. C. Benzmüller, F. Theiss, L. Paulson, and A. Fietzke. LEO-II - a cooperative automatic theorem prover for higher-order logic. In *Proc. of IJCAR 2008*, number 5195 in LNAI, pages 162–170. Springer, 2008.
13. C. Benzmüller and B. Woltzenlogel Paleo. Formalization, Mechanization and Automation of Gödel's Proof of God's Existence. *ArXiv e-prints*, 2013.

14. Christoph Benzmüller. Verifying the modal logic cube is an easy task (for higher-order automated reasoners). In Simon Siegler and Nathan Wasser, editors, *Verification, Induction, Termination Analysis - Festschrift for Christoph Walther on the Occasion of His 60th Birthday*, volume 6463 of *LNCS*, pages 117–128. Springer, 2010.
15. Christoph Benzmüller and Chad Brown. The curious inference of Boolos in MIZAR and OMEGA. In Roman Matuszewski and Anna Zalewska, editors, *From Insight to Proof – Festschrift in Honour of Andrzej Trybulec*, volume 10(23) of *Studies in Logic, Grammar, and Rhetoric*, pages 299–388. The University of Bialystok, Polen, 2007.
16. Christoph Benzmüller, Chad E. Brown, and Michael Kohlhase. Higher-order semantics and extensionality. *J. Symb. Log.*, 69(4):1027–1088, 2004.
17. Christoph Benzmüller, Jens Otten, and Thomas Raths. Implementing and evaluating provers for first-order modal logics. In Luc De Raedt, Christian Bessiere, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter Lucas, editors, *ECAI 2012*, volume 242 of *Frontiers in Artificial Intelligence and Applications*, pages 163–168, Montpellier, France, 2012. IOS Press.
18. Christoph Benzmüller and Bruno Woltzenlogel Paleo. Gödel's God in Isabelle/HOL. *Archive of Formal Proofs*, 2013, 2013.
19. Christoph Benzmüller and Bruno Woltzenlogel Paleo. Automating Gödel's ontological proof of God's existence with higher-order automated theorem provers. In Torsten Schaub, Gerhard Friedrich, and Barry O'Sullivan, editors, *ECAI 2014*, volume 263 of *Frontiers in Artificial Intelligence and Applications*, pages 93 – 98. IOS Press, 2014.
20. Christoph Benzmüller and Lawrence Paulson. Exploring properties of normal multimodal logics in simple type theory with LEO-II. In Christoph Benzmüller, Chad Brown, Jörg Siekmann, and Richard Statman, editors, *Reasoning in Simple Type Theory — Festschrift in Honor of Peter B. Andrews on His 70th Birthday*, Studies in Logic, Mathematical Logic and Foundations, pages 386–406. College Publications, 2008.
21. Christoph Benzmüller and Lawrence Paulson. Quantified multimodal logics in simple type theory. *Logica Universalis (Special Issue on Multimodal Logics)*, 7(1):7–20, 2013.
22. Christoph Benzmüller and Adam Pease. Higher-order aspects and context in SUMO. *Journal of Web Semantics (Special Issue on Reasoning with context in the Semantic Web)*, 12-13:104–117, 2012.
23. Christoph Benzmüller and Thomas Raths. HOL based first-order modal logic provers. In Kenneth L. McMillan, Aart Middeldorp, and Andrei Voronkov, editors, *Proceedings of the 19th International Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR)*, volume 8312 of *LNCS*, pages 127–136, Stellenbosch, South Africa, 2013. Springer.
24. Christoph Benzmüller, Leon Weber, and Bruno Woltzenlogel Paleo. Computer-assisted analysis of the Anderson-Hájek ontological controversy. In Ricardo Souza Silvestre and Jean-Yves Béziau, editors, *Handbook of the 1st World Congress on Logic and Religion, Joao Pessoa, Brasil*, 2015.
25. J.Y. Beziau, W. Carnielli, and D. Gabbay, editors. *Handbook of Paraconsistency*. College Publications, 2007.
26. Patrick Blackburn, Johan F. A. K. van Benthem, and Frank Wolter. *Handbook of Modal Logic, Volume 3 (Studies in Logic and Practical Reasoning)*. Elsevier Science Inc., New York, NY, USA, 2006.
27. J.C. Blanchette, S. Böhme, and L.C. Paulson. Extending Sledgehammer with SMT solvers. *Journal of Automated Reasoning*, 51(1):109–128, 2013.
28. J.C. Blanchette and T. Nipkow. Nitpick: A counterexample generator for higher-order logic based on a relational model finder. In *Proc. of ITP 2010*, number 6172 in LNCS, pages 131–146. Springer, 2010.

29. George Boolos. A curious inference. *Journal of Philosophical Logic*, 16:1–12, 1987.
30. C.E. Brown. Satallax: An automated higher-order prover. In *Proc. of IJCAR 2012*, number 7364 in LNAI, pages 111 – 117. Springer, 2012.
31. S. Bucav, V. Buvac, and I.A. Mason. Metamathematics of contexts. *Fundamenta Informaticae*, 23(3):263–301, 1995.
32. Arthur Charguéraud. The locally nameless representation. *J. Autom. Reasoning*, 49(3):363–408, 2012.
33. R. Corazzon. Contemporary bibliography on ontological arguments: `http://www.ontology.co/biblio/ontological-proof-contemporary-biblio.htm`.
34. N.C.A. da Costa and E.H. Alves. Semantical analysis of the calculi cn. *Notre Dame Journal of Formal Logic*, 18(4):621–630, 1977.
35. J.M. Dunn and G. Restall. Relevance logic. *Handbook of Philosophical Logic*, 6:1–136, 2002.
36. M. Fitting. *Types, Tableaux and Gödel's God*. Kluwer, 2002.
37. M. Fitting and R.L. Mendelsohn. *First-Order Modal Logic*, volume 277 of *Synthese Library*. Kluwer, 1998.
38. Dov M. Gabbay. *Labelled Deductive Systems*. Clarendon Press, 1996.
39. D. Gallin. *Intens. and Higher-Order Modal Logic*. N.-Holland, 1975.
40. F. Giunchiglia. Contextual reasoning. *Epistemologia (Special Issue on Languages and Machines)*, 16:345—-364, 1993.
41. F. Giunchiglia and L. Serafini. Multilanguage hierarchical logics or: How we can do without modal logics. *Artificial Intelligence*, 65(1):29–70, 1994.
42. K. Gödel. Ontological proof. In *Kurt Gödel: Collected Works Vol. 3: Unpublished Essays and Letters*. Oxford University Press, 1970.
43. K. Gödel. *Appx.A: Notes in Kurt Gödel's Hand*, pages 144–145. In [70], 2004.
44. R. V. Guha. *Context: A Formalization and Some Applications*. PhD thesis, Stanford University, 1991.
45. L. Henkin. Completeness in the theory of types. *Journal of Symbolic Logic*, 15(2):81–91, 1950.
46. S. Jaśkowski. Rachunek zdań dla systemów dedukcyjnych sprzecznych. *Studia Societatis Scientiarun Torunesis*, 1(5):55–77, 1948.
47. S. Jaśkowski. Propositional calculus for contradictory deductive systems. *Studia Logica*, 24:143–157, 1969.
48. Cezary Kaliszyk and Josef Urban. Hol(y)hammer: Online ATP service for HOL light. *Mathematics in Computer Science*, 9(1):5–22, 2015.
49. Cezary Kaliszyk and Josef Urban. Learning-assisted theorem proving with millions of lemmas. *J. Symb. Comput.*, 69:109–128, 2015.
50. Fredrik Lindblad. agsyHOL website. `https://github.com/frelindb/agsyHOL`.
51. John McCarthy. Generality in artificial intelligence. *Communications of the ACM*, 30(12):1030–1035, 1987.
52. John McCarthy. Notes on formalizing context. In *Proceedings of IJCAI'93*, pages 555–562, 1993.
53. R. Muskens. Higher Order Modal Logic. In P Blackburn et al., editor, *Handbook of Modal Logic*, pages 621–653. Elsevier, Dordrecht, 2006.
54. T. Nipkow, L.C. Paulson, and M. Wenzel. *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*. Number 2283 in LNCS. Springer, 2002.
55. Jens Otten. Mleancop: A connection prover for first-order modal logic. In Stéphane Demri, Deepak Kapur, and Christoph Weidenbach, editors, *Automated Reasoning - 7th International Joint Conference, IJCAR 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 19-22, 2014. Proceedings*, volume 8562 of *Lecture Notes in Computer Science*, pages 269–276. Springer, 2014.

56. B. Woltzenlogel Paleo and C. Benzmüller. Formal theology repository (https://github.com/FormalTheology/GoedelGod).
57. C. Paulin-Mohring. Introduction to the calculus of inductive constructions. In D. Delahaye and B. Woltzenlogel Paleo, editors, *All about Proofs, Proofs for All*, Mathematical Logic and Foundations. College Publications, London, 2015.
58. Adam Pease. *Ontology: A Practical Guide*. Articulate Software Press, 2011.
59. Adam Pease, editor. *Ontology: A Practical Guide*. Articulate Software Press, Angwin, CA 94508, 2011.
60. Adam Pease and Geoff Sutcliffe. First order reasoning on a large ontology. In G. Sutcliffe, J. Urban, and S. Schulz, editors, *Proceedings of the CADE-21 Workshop on Empirically Successful Automated Reasoning in Large Theories (ESARLT)*, volume 257 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.
61. G. Priest. Paraconsistent belief revision. *Theoria*, 67:214 – 228, 2001.
62. G. Priest and R. Sylvan. Simplified semantics for basic relevant logics. *Journal of Philosophical Logic*, 1992.
63. Deepak Ramachandran, Pace Reagan, and Keith Goolsbey. First-orderized ResearchCyc: Expressivity and efficiency in a common-sense ontology. In Shvaiko P., editor, *Papers from the AAAI Workshop on Contexts and Ontologies: Theory, Practice and Applications, Pittsburgh, Pennsylvania, USA, 2005*. Technical Report WS-05-01 published by The AAAI Press, Menlo Park, California, July 2005.
64. T. Raths and J. Otten. The QMLTP problem library for first-order modal logics. In *Proc. of IJCAR 2012*, volume 7364 of *LNCS*, pages 454–461. Springer, 2012.
65. G. Restall and J. Slaney. Realistic belief revision. In *Proceedings of the Second World Conference in the Fundamentals of Artificial Intelligence*, pages 367–378, 1995.
66. D. Scott. *Appx.B: Notes in Dana Scott's Hand*, pages 145–146. In [70], 2004.
67. Luciano Serafini and Paolo Bouquet. Comparing formal theories of context in AI. *Artificial Intelligence*, 155:41–67, May 2004.
68. A. Siders and B. Woltzenlogel Paleo. A variant of Gödel's ontological proof in a natural deduction calculus. (github.com/FormalTheology/GoedelGod/blob/master/Papers/InProgress/NaturalDeduction/GodProof-ND.pdf?raw=true).
69. J.H. Sobel. Gödel's ontological proof. In *On Being and Saying. Essays for Richard Cartwright*, pages 241–261. MIT Press, 1987.
70. J.H. Sobel. *Logic and Theism: Arguments for and Against Beliefs in God*. Cambridge U. Press, 2004.
71. G. Sutcliffe. The TPTP problem library and associated infrastructure. *Journal of Automated Reasoning*, 43(4):337–362, 2009.
72. Geoff Sutcliffe and Christoph Benzmüller. Automated reasoning in higher-order logic using the TPTP THF infrastructure. *Journal of Formalized Reasoning*, 3(1):1–27, 2010.
73. K. Tanaka. Three schools of paraconsistency. *The Australasian Journal of Logic*, 2003.
74. Makarius Wenzel. Hoare logic in isabelle. http://isabelle.in.tum.de/dist/library/HOL/HOL-Isar_Examples/Hoare.html.