Sweet *SIXTEEN*: Automation via Embedding into Classical Higher-Order Logic

Alexander Steen   Christoph Benzmüller

Abstract

An embedding of many-valued logics based on *SIXTEEN* in classical higher-order logic is presented. The theoretical motivation is to demonstrate that many-valued logics, like many other non-classical logics, can be elegantly modeled (and even combined) as fragments of classical higher-order logic. Equally relevant are the pragmatical aspects of the presented approach: interactive and automated reasoning in many valued logics, which have many applications in computer science, artificial intelligence, linguistics, philosophy and mathematics, is readily enabled with state of the art reasoning tools for classical higher-order logic.

Keywords. many-valued logic; non-classical logic; higher-order logic; automated theorem proving; semantical embedding; automation; meta-logical reasoning

1 Introduction

Classical logics are based on the bivalence principle, that is, the set of truth-values V has cardinality $|V| = 2$, usually with $V = \{\mathrm{T}, \mathrm{F}\}$ where T and F stand for truthhood and falsity, respectively. Many-valued logics (MVL) generalize this requirement and allow V to be a more or less arbitrary set of truth-values, often referred to as *truth-degrees*. Popular examples of many-valued logics are fuzzy logics [**?**, **?**] with an uncountable set of truth-degrees, Gödel logics [**?**, **?**] and Łukasiewicz logics [22] with denumerable sets of truth-degrees, and, from the class of finitely-many-valued logics, Dunn/Belnap's four-valued logic [5, 6].

The latter system, although originating from research on relevance logics, has been of strong interest to computer scientists as formal foundation of information and knowledge bases. Here, the set of truth-degrees is given by the power set of $\{\mathrm{T}, \mathrm{F}\}$, i.e. $V = \{\mathbf{N}, \mathbf{T}, \mathbf{F}, \mathbf{B}\}$, where $\mathbf{N}$ denotes the empty set $\emptyset$ (mnemonic for *None*), $\mathbf{B}$ the set $\{\mathrm{T}, \mathrm{F}\}$ (for *Both*), and $\mathbf{T}$ and $\mathbf{F}$ denote the singleton sets containing the respective classical truth-value.

This article presents an approach for automating MVL based on a sixteen-valued lattice, denoted *SIXTEEN* [26]. This system has been developed by Shramko and Wansing as a generalization of the mentioned Dunn/Belnap four-valued system to knowledge bases in computer networks [25] and was subsequently further investigated in various contexts (e.g. [24, 26]). In *SIXTEEN*, the truth-degrees are given by the power set of Dunn/Belnap's truth values, i.e. $V = 2^{\{\mathbf{N},\mathbf{T},\mathbf{F},\mathbf{B}\}}$. This generalization is essentially motivated by the observation that a four-valued system cannot express certain phenomena that arise in knowledge bases in computer networks. Further applications in linguistics and philosophy are discussed in the monograph by Shramko and Wansing [26], to which we refer to for a thorough investigation.

While the use of MVL, in particular SIXTEEN, for knowledge representation and reasoning in computer science, linguistics and philosophy is well justified, there are unfortunately no tools available yet that support automated or interactive reasoning in SIXTEEN. This applies also to most other MVL systems (and the number of available systems significantly decreases for quantified MVLs).

To that end, we present a semantical embedding of logics based on *SIXTEEN* within classical higher-order logic (HOL). Using this encoding, ordinary higher-order automated theorem provers can be exploited for reasoning within the many-valued setting of *SIXTEEN*. In addition, due to the expressivity of the host language, automation of meta-logical reasoning is included for free.

The semantical embedding approach provides similar results for other non-classical logics, yielding out-of-the-box automation of many other logics using ordinary HOL reasoning systems. Most recent related work has been done in the context of automation of higher-order quantified modal logic [12, 14], quantified conditional logics [10] and quantified hybrid logics [27]. There is empirical evidence that such tools can be employed to successfully verify or refute non-trivial in metaphysics and that they even contribute some new knowledge [**?**, 13, 8].

The remainder of this article is organized as follows: In §2, the above mentioned logics based on *SIXTEEN* are introduced. §3 and §4 address HOL and its utilization for automating reasoning within MVL. Subsequently, in §5 experiments with the aforementioned encoding are displayed and discussed. Finally, §6 concludes the article and sketches further extensions of the presented approach.

2 Many-Valued Logics Based on *SIXTEEN*

The MVL systems we will be addressing here are, as described earlier in §1, based on a sixteen-valued structure of truth-degrees. The underlying set V of truth-degrees is hereby given by the power set of the power set of the classical (bivalent) truth-values $\{\mathrm{T},\mathrm{F}\}$, i.e. $V := 2^{2^{\{\mathrm{T},\mathrm{F}\}}}$. The set V thus further generalizes the set of truth-degrees of Dunn/Belnap's system. More precisely, we have

$$V = 2^{\{\mathbf{N},\mathbf{T},\mathbf{F},\mathbf{B}\}} = \{\mathbf{N}, \boldsymbol{N}, \boldsymbol{T}, \boldsymbol{F}, \boldsymbol{B}, \mathbf{NT}, \ldots, \mathbf{NTFB}\}$$

where $\boldsymbol{N}, \boldsymbol{T}, \boldsymbol{F}$ and $\boldsymbol{B}$ are the respective singleton sets containing $\mathbf{N}, \mathbf{T}, \mathbf{F}$ and $\mathbf{B}$. The remaining truth-degrees are named using a combination of the letters $\mathbf{N}, \mathbf{T}, \mathbf{F}$ and $\mathbf{B}$, representing the truth-degree that contains the respective elements when regarded as a set (e.g. $\mathbf{NT}$ for the set $\{\mathbf{N}, \mathbf{T}\}$).

Using the above set V, there are multiple (in fact, mutually independent) possibilities on how to order the truth-degrees in a meaningful way. They can, for instance, be sorted by increasing truth. But there are other reasonable orderings one can think of, e.g. when interested in the decrease of falsity (which is not the same thing as increase of truth).

Shramko and Wansing [26, pp. 53–57] suggest three reasonable independent (partial) orderings, for the set of truth-degrees V.

First, the ordering $\leq_i$ orders elements of V by *information*. Here, a truth-degree v is smaller than w with respect to its information value, if and only if v is a subset of w, i.e. $v \leq_i w :\Leftrightarrow v \subseteq w$. The remaining two orderings which are more suited for logical reasoning are $\leq_t$ and $\leq_f$, comparing truth-degrees by their *truth* and *falsity*, respectively. For a formal definition of these orderings, we need to introduce the notion of "truthful" and "truthless" subsets of a truth-degree v: The truthful subset of v, denoted v^t, contains exactly those elements in v which themselves contain T. The truthless subset of v, denoted v^{-t}, accordingly consists only of those elements of v which do not contain T. This notion is analogously extended to $(.)^f$ and $(.)^{-f}$. More formally, we have

$$\begin{aligned} v^t &:= \{x \in v \mid \mathrm{T} \in x\} \\ v^{-t} &:= \{x \in v \mid \mathrm{T} \notin x\} \end{aligned} \tag{1}$$

and for its counterpart based on falsity

$$\begin{aligned} v^f &:= \{x \in v \mid \mathrm{F} \in x\} \\ v^{-f} &:= \{x \in v \mid \mathrm{F} \notin x\} \end{aligned} \tag{2}$$

Note that we have $v^t \neq v^{-f}$ and $v^f \neq v^{-t}$, i.e. the two respective counterparts of these sets do not coincide. As it is pointed out by Shramko and Wansing, the counterparts of this notions do indeed coincide for the four-valued system of Dunn/Belnap [26, p.53]. That is why there is a single unique *logical ordering* in that system, as opposed to the system described here.

The ordering $\leq_t$ can elegantly been defined as an increase in truth and a non-increase of non-truth. Analogously, $\leq_f$ orders by increase of falsity and non-increase of non-falsity:

$$\begin{aligned} v \leq_t w &:\Leftrightarrow v^t \subseteq w^t \wedge w^{-t} \subseteq v^{-t} \\ v \leq_f w &:\Leftrightarrow v^f \subseteq w^f \wedge w^{-f} \subseteq v^{-f} \end{aligned} \tag{3}$$

All the above orderings $\leq_i, \leq_t$ and $\leq_f$ induce a so-called *trilattice*

$$\mathit{SIXTEEN} = (V, \sqcup_i, \sqcap_i, \sqcup_t, \sqcap_t, \sqcup_f, \sqcap_f)$$

which is essentially a threefold lattice, i.e. having three mutually independent pairs of meet and join operations.

Additionally to the above meet and join operations, there is an inversion operation, denoted by $-_\Box$, for each axis $\Box \in \{t, f, i\}$ of the trilattice. An important property of the inversion for a specific axis is that it does not change the order with respect to the other axes. For instance, if $x \leq_t y$, then $(-_t\, y) \leq_t$

$(-_t\ x)$, but still $(-_f\ x) \leq_t (-_f\ y)$ and $(-_i\ x) \leq_t (-_i\ y)$, i.e. ordering by truth is invariant under f-inversion and i-inversion.

We are now sufficiently prepared to present the syntax and semantics for respective logics based on truth- and falsity-orderings. The three logics studied in the remainder are denoted as $\mathcal{L}_t$, $\mathcal{L}_f$, and $\mathcal{L}_{tf}$. Their abstract syntax is given as:

$$\begin{aligned}\mathcal{L}_t:\ & A, B ::= x \mid A \wedge_t B \mid A \vee_t B \mid\ \sim_t A\\ \mathcal{L}_f:\ & A, B ::= x \mid A \wedge_f B \mid A \vee_f B \mid\ \sim_f A\end{aligned}$$

where x is a propositional variable, and $\wedge$, $\vee$, and $\sim$ are the respective conjunction, disjunction and negation. We will primarily focus on $\mathcal{L}_{tf}$ since the other languages are proper fragments of it.

To provide a semantics, let v^{16} be a *16-valuation*, that is, a map from propositional variables to the sixteen-valued set V. The semantic evaluation of propositional variables is extended to compound formulate as usual ($\Box \in \{t, f\}$):

$$\begin{aligned} v^{16}(A \wedge_\Box B) &:= v(A) \sqcap_\Box v(B)\\ v^{16}(A \vee_\Box B) &:= v(A) \sqcup_\Box v(B)\\ v^{16}(\sim_\Box A) &:= -_\Box v(A)\end{aligned} \tag{4}$$

Semantic entailment can now be defined as an increase in truth or as an decrease in falsity. More formally, for two arbitrary formulas $A, B \in \mathcal{L}_{tf}$, A entails B wrt. to truth order, $A \models_t^{16} B$, if and only if $v^{16}(A) \leq_t v^{16}(B)$ for all 16-valuations v^{16}. Analogously we have $A \models_f^{16} B$ if and only if $v^{16}(B) \leq_f v^{16}(A)$, for all 16-valuations v^{16}. The resulting logics are $(\mathcal{L}_{tf}, \models_t^{16})$, $(\mathcal{L}_{tf}, \models_f^{16})$ and the *bi-consequence logic* $(\mathcal{L}_{tf}, \models_t^{16}, \models_f^{16})$ [26, p. 65].

3 Classical Higher-Order Logic

Higher-order logic (HOL) is an elegant and expressive formal system that extends first-order logic with quantification over arbitrary sets and functions. Church [18] proposed a version of higher-order logic, called simple type theory (in the following referred to as HOL), which he built on top of the simply typed λ-calculus [16, 17]. The simply typed λ-calculus augments the untyped λ-calculus, as studied by Alonzo Church in the 1930s, with *simple types.* The set of simple types $\mathcal{T}$ is thereby freely generated from a set of base types and a function type constructor. In HOL, the set of base types is usually taken as (a superset of) $\{\iota, o\}$ with ι and o being the type of individuals and classical truth values, respectively.

The terms of the logic are essentially those of the simply typed λ-calculus, enriched with typed (logical) constants. These constants are taken from a family of denumerable sets of constant symbols $\Sigma := (\Sigma_\tau)_{\tau\in\mathcal{T}}$, called *signature.* Together with a family of typed variable symbols $(\mathcal{V}_\tau)_{\tau\in\mathcal{T}}$ the terms of HOL are then those terms contained in the smallest set Λ for which the following conditions hold: Each constant symbol $c_\tau \in \Sigma_\tau$ and each variable symbol $X_\tau \in \mathcal{V}_\tau$

is a HOL term of type τ. If $X_\tau \in \mathcal{V}_\tau$ is a variable symbol and $s_\nu \in \Lambda$ is a HOL term, then the *abstraction* (of s_ν) $(\lambda X_\tau.\ s_\nu)_{\nu\tau} \in \Lambda$ is a HOL term of type $\nu\tau$. Finally, if $s_\tau, t_{\nu\tau} \in \Lambda$ are HOL terms, then the *application* (of $t_{\nu\tau}$ onto s_τ) given by $(t_{\nu\tau}\ s_\tau)_\nu \in \Lambda$ is a HOL term of type ν. Hereby $\tau, \nu \in \mathcal{T}$ are types and the *abstraction type* $\nu\tau$ denotes the type of functions from arguments of type τ to values of type ν. Abstraction types are considered left-associative, i.e. $\tau\nu\mu \equiv (\tau\nu)\mu$. As usual for Church-style typing, a term's type is given as subscript and considered a part of its name, hence intrinsic to it. Nevertheless, we may omit type subscripts in the following if clear from the context.

We choose the signature Σ to consist at least of the *primitive logical connectives*, that are negation $\neg_{oo}$, disjunction $\vee_{ooo}$, and universal quantification $\Pi^\tau_{o(o\tau)}$ for each type $\tau \in \mathcal{T}$. The remaining (non-primitive) logical connectives can be defined as abbreviations in the usual way.

We use *binder notation* $\forall X_\tau.\ s_o$ as shorthand for universal quantification given by $\Pi^\tau_{(\tau\to o)\to o}(\lambda X_\tau.\ s_o)$. For additional convenience, we allow infix notation for the common binary logical connectives, i.e. write $(s \vee t)$ instead of $((\vee\ s)\ t)$.

A *formula* of HOL is a term $s_o \in \Lambda$, hence of type o. As usual, a *sentence* is a closed formula.

The usual rules of λ-conversions (α-, β-, and η-conversion) are intrinsically included in HOL. Using these conversions, especially β-reduction, all quantifier instantiations can be expressed very concisely.

The meta-operation of *substituting* a variable X_τ by some term t_τ in s_ν is denoted $s[X_\tau/t_\tau]$. Hereby, we assert that there is no variable capture happening in s_τ by assuming α-conversion as implicit when necessary. A β-redex of the form $(\lambda X_\tau.\ s_\nu)\ t_\tau$ then β-reduces to $s_\nu[X_\tau/t_\tau]$. A term s_τ is said to be in β-normal form if it does not contain any β-redex as subterm. The β-normal form of s_τ is denoted $s_\tau{\downarrow_\beta}$, equivalence modulo β-conversion (and α-conversion) is denoted $=_\beta$. Reduction, normal forms and equivalence modulo η and $\beta\eta$ are defined analogously. We refer to the literature for a thorough study of typed λ-calculi [4].

The semantics of HOL is meanwhile well-understood [11] and various semantic generalizations have been studied: We here summarize the most important points: As a consequence of Gödel's incompleteness theorem [19], the so-called *standard semantics* of HOL is necessarily incomplete. However, it shows that in many practical applications Henkin's weaker form of *general semantics* [20, 2, 1] is sufficiently expressive. For Henkin's generalized semantics sound and and complete proof calculi exists. And such proof calculi provide the theoretical foundations of modern theorem provers for HOL such as Leo-II [9] and Satallax [15].

Standard and Henkin semantics is introduced more formally next. We start out with the notion of frames.

A *frame* is a collection $\{\mathcal{D}_\tau\}_{\tau\in\mathcal{T}}$ of non-empty domains with a fixed set of classical truth-values $\mathcal{D}_o = \{T, F\}$ (for truth and falsehood respectively) and sets $\mathcal{D}_{\nu\tau}$ which is the domain of functions from $\mathcal{D}_\tau$ to $\mathcal{D}_\nu$.

A *model* M is a pair $M = (\{\mathcal{D}_\tau\}_{\tau\in\mathcal{T}}, I)$, where I is a function that maps each constant symbol $c_\tau \in \Sigma_\tau$ to an element of $\mathcal{D}_\tau$ (the *denotation of* c_τ). The function I is chosen such that the logical connectives $\neg_{oo}$, $\vee_{ooo}$ and $\Pi_{o(o\alpha)}$ have their usual meaning.

A variable assignment g is a map that assigns each variable $X_\tau \in \mathcal{V}_\tau$ an element in $\mathcal{D}_\tau$. With $g[Y/s]$ we mean the variable assignment that is identical to g except that variable Y is now mapped to s.

Finally, given a model $M = (\{\mathcal{D}_\tau\}_{\tau\in\mathcal{T}}, I)$ and a variable assignment g, the *value* of a HOL term (with respect to M and g), denoted by $\|.\|^{M,g}$, is given by

(i) $\|X_\tau\|^{M,g} = g(X_\tau)$ and $\|c_\tau\|^{M,g} = I(c_\tau)$

(ii) $\|(s_{\nu\tau}\ t_\tau)\|^{M,g} = \|s_{\nu\tau}\|^{M,g}\ \|t_\tau\|^{M,g}$

(iii) $\|(\lambda X_\tau.\ s_\nu)\|^{M,g}$ is a function $f \in \mathcal{D}_{\nu\tau}$ s.t. for all $z \in \mathcal{D}_\tau$ it holds that $f(z) = \|s_\nu\|^{M,g[X_\tau/z]}$

A model M is called *standard model* if and only if the sets $\mathcal{D}_{\nu\tau}$ are chosen to be the complete set $\mathcal{D}_\nu^{\mathcal{D}_\tau}$ of function from domain τ to co-domain ν. The notion of general models (or *Henkin models*) is, in contrast, defined by choosing $\mathcal{D}_{\nu\tau}$ as as subset of $\mathcal{D}_\nu^{\mathcal{D}_\tau}$ such that it contains "sufficiently many", but not necessarily all, functions. More formally, M is a general model if and only if $\|.\|^{M,g}$ is a total function (that is, every term is assigned a value). The function $\|.\|^{M,g}$ is uniquely determined for every general model. Of course, every standard model is also a general model.

For a model M and a variable assignment g, a formula s_o is *true in model M wrt. variable assignment g*, denoted $M, g \models^{HOL} s_o$, if $\|s_o\|^{M,g} = T$. It is called *valid in M* if $M, g \models^{HOL} s_o$ for all variable assignments g. This is written as $M \models^{HOL} s_o$. Finally, a formula s_o is called *Henkin-valid* (or simply *valid*), written $\models^{HOL} s_o$, if s_o is valid in every Henkin model. In the following, we always assume general semantics of HOL.

4 Embedding of $\mathcal{L}_{tf}$

In this section we present a semantical embedding of the logic $\mathcal{L}_{tf}$ – and thereby automatically also for $\mathcal{L}_t$ and $\mathcal{L}_f$ – in HOL. The idea is essentially to exploit the expressiveness of HOL for encoding the semantics of the given truth-degrees and the operations on them.

The truth-degrees of *SIXTEEN* have been introduced as sets of sets of (classical) truth-values T and F (cf. §2). We can elegantly represent sets via characteristic functions in HOL and utilize λ-abstraction for this purpose. Exploiting this idea we below encode all sixteen sets that match the respective truth-degree. More precisely, a set $M = \{x \mid P(x)\}$ is modeled here by its *characteristic function* $\chi_M = (\lambda x.P(x))$, which is a predicate that holds for any element m contained in M and does not hold for any other element $m \notin M$. These λ-abstractions are typed in HOL. For example, the characteristic function for a set of truth values, $(\lambda x_o.P(x))$, has type oo. Consequently, the characteristic function for a set of sets of truth values has type $o(oo)$. Thus, truth-degrees of *SIXTEEN* correspond to functions of type $o(oo)$. A single truth-degree is then a function $(\lambda n_{oo}.\ P(n))$ where $P(n)$ is an explicit predicate on n that via function application determines which elements are to be contained within (the set) n so that n is itself contained in the set (of sets) under consideration. In other words the sets as characteristic functions approach is applied here in nested fashion.

As an example, consider the truth-degree $\boldsymbol{N}$, which corresponds to the set $\{\emptyset\}$, i.e. the set only containing the empty set of truth values. Note that this set $\boldsymbol{N}$ contains exactly those sets of truth values that neither contain T nor F; the empty set of truth values is hence the sole candidate fulfilling this condition. Consequently, our encoding of $\boldsymbol{N}$ is $(\lambda n_{oo}.\ \neg(n\ \mathrm{F}) \wedge \neg(n\ \mathrm{T}))$.

$\mathbf{N}$	$=$	$\lambda n_{oo}.\ \mathrm{F}$
$\boldsymbol{N}$	$=$	$\lambda n_{oo}.\ \neg(n\ \mathrm{F}) \wedge \neg(n\ \mathrm{T})$
$\boldsymbol{T}$	$=$	$\lambda n_{oo}.\ \neg(n\ \mathrm{F}) \wedge n\ \mathrm{T}$
$\boldsymbol{F}$	$=$	$\lambda n_{oo}.\ n\ \mathrm{F} \wedge \neg(n\ \mathrm{T})$
$\boldsymbol{B}$	$=$	$\lambda n_{oo}.\ n\ \mathrm{F} \wedge n\ \mathrm{T}$
$\mathbf{NF}$	$=$	$\lambda n_{oo}.\ \neg(n\ \mathrm{T})$
$\mathbf{NT}$	$=$	$\lambda n_{oo}.\ \neg(n\ \mathrm{F})$
$\mathbf{NB}$	$=$	$\lambda n_{oo}.\ (\neg(n\ \mathrm{F}) \wedge \neg(n\ \mathrm{T})) \vee (n\ \mathrm{F} \wedge n\ \mathrm{T})$
$\mathbf{FT}$	$=$	$\lambda n_{oo}.\ (n\ \mathrm{F} \wedge \neg(n\ \mathrm{T})) \vee (\neg(n\ \mathrm{F}) \wedge n\ \mathrm{T})$
$\mathbf{FB}$	$=$	$\lambda n_{oo}.\ n\ \mathrm{F}$
$\mathbf{TB}$	$=$	$\lambda n_{oo}.\ n\ \mathrm{T}$
$\mathbf{NFT}$	$=$	$\lambda n_{oo}.\ \neg(n\ \mathrm{F}) \vee \neg(n\ \mathrm{T})$
$\mathbf{NFB}$	$=$	$\lambda n_{oo}.\ n\ \mathrm{F} \vee \neg(n\ \mathrm{T})$
$\mathbf{NTB}$	$=$	$\lambda n_{oo}.\ \neg(n\ \mathrm{F}) \vee n\ \mathrm{T}$
$\mathbf{FTB}$	$=$	$\lambda n_{oo}.\ n\ \mathrm{F} \vee n\ \mathrm{T}$
$\mathbf{A}$	$:=$	$\mathbf{NFTB} = \lambda n_{oo}.\ \mathrm{T}$

Table 1: Encoding of all truth-degrees of *SIXTEEN* in HOL. Types are omitted where possible.

We now present the encoding of the logical operations of $\mathcal{L}_{tf}$. Recall that their semantics is defined using the lattice operations $\sqcup, \sqcap$ and the inversion operation $-$ as introduced in §2.

Again, the notion of truthful subsets $(.)^t$ and truthless subsets $(.)^{-t}$ (cf. Eq. (1)) is needed to define the ordering $\leq_t$ on truth-degrees:

$$(v)^t_{o(oo)} := \lambda n_{oo}.\ (v\ n) \wedge (n\ \mathrm{T}) \qquad (v)^{-t}_{o(oo)} := \lambda n_{oo}.\ (v\ n) \wedge \neg(n\ \mathrm{T})$$

For any truth degree v, $(v)^t$ is itself again a set of sets of truth-values, hence its encoding is similar to that of truth-degrees. Here, the sub-expression $(v\ n)$ asserts that n is contained in v, and $(n\ \mathrm{T})$ ensures that T is contained in n. The analogous embedding of Eq. 2 is given by

$$(v)^f_{o(oo)} := \lambda n_{oo}.\ (v\ n) \wedge (n\ \mathrm{F}) \qquad (v)^{-f}_{o(oo)} := \lambda n_{oo}.\ (v\ n) \wedge \neg(n\ \mathrm{F})$$

The orderings $\leq_t$ and $\leq_f$ can then be encoded to match the definition of Eq. 3 using the same techniques as before:

$$\leq_t := \lambda v_{o(oo)}.\lambda w_{o(oo)}.\forall n_{oo}.\ ((v^t\ n) \Rightarrow (w^t\ n)) \wedge ((w^{-t}\ n) \Rightarrow (v^{-t}\ n))$$
$$\leq_f := \lambda v_{o(oo)}.\lambda w_{o(oo)}.\forall n_{oo}.\ ((v^f\ n) \Rightarrow (w^f\ n)) \wedge ((w^{-f}\ n) \Rightarrow (v^{-f}\ n))$$

The embedding of $\sqcup, \sqcap$ and $-$ are slightly more complicated as we need a closed algebraic description for these operators. In the original description, only an implicit characterization via properties is given for each of these operations. Up to the authors' knowledge, there has not been any such closed algebraic formulation in the literature. As it turns out, the join and meet operations can be defined as

$$\sqcup_t := \lambda v_{o(oo)}.\lambda w_{o(oo)}.v^t \cup w^t \cup \left(w^{-t} \cap v^{-t}\right)$$
$$\sqcap_t := \lambda v_{o(oo)}.\lambda w_{o(oo)}.v^{-t} \cup w^{-t} \cup \left(w^t \cap v^t\right)$$
$$\sqcup_f := \lambda v_{o(oo)}.\lambda w_{o(oo)}.v^f \cup w^f \cup \left(w^{-f} \cap v^{-f}\right)$$
$$\sqcap_f := \lambda v_{o(oo)}.\lambda w_{o(oo)}.v^{-f} \cup w^{-f} \cup \left(w^f \cap v^f\right)$$

The intuition behind these definitions is as follows: Join operations (here for $\sqcup_t$) construct a set that combines the ”truthful” elements of the truth-degree while only containing those ”truthless’ elements that were contained in both sets. That is compatible with the ordering idea of $\leq_t$, where bigger elements increase $(.)^t$ while not-increasing $(.)^{-t}$. A similar argumentation holds for the meet operations, yielding smaller elements with respect to the respective ordering.

Finally, the inversion operation $-_t v$ can be encoded by explicitly constructing sets $(\lambda b_o. \cdots)$ for each element n of the original truth-degree v such that it contains T whenever n does not contain T, and it contains F if and only if F is contained in n. That way we only swap the property whether an element of v contained T, hence inverting it with respect to $\leq_t$:

$$-_t := \lambda v_{o(oo)}.\lambda n_{oo}.\ (v\ (\lambda b_o.(\neg b \Rightarrow n\ \mathrm{F}) \wedge (b \Rightarrow \neg(n\ \mathrm{T}))))$$

An analogous construction is employed for $-_f v$, where elements of v containing F are swapped for elements that do not contain F but still contain T if they originally did:

$$-_f := \lambda v_{o(oo)}.\lambda n_{oo}.\ (v\ (\lambda b_o.(\neg b \Rightarrow \neg(n\ \mathrm{F})) \wedge (b \Rightarrow n\ \mathrm{T})))$$

Semantical entailment $\models^{16}$ can already be expressed by the above definitions because it is defined via increase of truth (or decrease of falsity), i.e. by means of $\leq_t$ and $\leq_f$.

Soundness and Completeness

Theorem 4.1. *Let A, B be $\mathcal{L}_{tf}$ formulas and let A^*, B^* be the corresponding embedded formulas in HOL according to our encoding from above. It holds*

$$A \models^{16}_t B :\Leftrightarrow v^{16}(A) \leq_t v^{16}(B) \quad \mathit{iff} \quad \models^{HOL} A^* \leq^*_t B^*$$

and

$$A \models^{16}_{f} B :\Leftrightarrow v^{16}(B) \leq_f v^{16}(A) \quad \textit{iff} \quad \models^{HOL} B^* \leq^*_f A^*$$

Proof. Remember that we identify characteristic functions and sets.

Step (1): Let $* \in V$. For all HOL model structures M and assignments g we have $\| * \|^{M,g} = *$.

Step (2): Let $* \in \{\sqcup_t, \sqcap_t, \sqcup_f, \sqcap_f, -_t, -_f\}$. For all HOL model structures M and assignments g we have $\| * \|^{M,g} = *$.

Step (3): Let $* \in \{v^t, v^f, v^{-t}, v^{-f}, \leq_t, \leq_f\}$. For all HOL model structures M and assignments g we have $\| * \|^{M,g} = *$.

The detailed proofs are straightforward. Now the theorem follows from (1)-(3) by a induction over the structure of A and B. □

5 Experiments and Results

To enable such experiments and further utilisation of the embedding in practice we have encoded the above embedding in TPTP THF syntax [28, 29], which is a concrete syntax format for HOL. An excerpt of this TPTP THF encoding is given in Fig. 1. Altogether this encoding consists of approx. 150 lines of code, including comments. It can simply be loaded as axiomatization file by any TPTP-compatible HOL ATP for reasoning within $\mathcal{L}_{tf}$. Additionally, we provided the embedding as a theory for the renowned interactive proof assistant Isabelle/HOL [23].

As a proof of concept for the practical usability of our automation approach, we formulated several proof tasks within and about the *SIXTEEN* system for ATP systems. The most interesting experiments hereby verify the correctness of our closed formulations and encoding of the lattice operations $\sqcup, \sqcap$ and $-$. To that end, we have checked its definitions against the appropriate properties given in the monograph of Shramko and Wansing [26, Prop. 3.2, Def. 3.6]. Table 2 displays the respective properties that has been given to ordinary higher-order ATP systems.

For our measurements, we used the two automated theorem provers LEO-II [9] and Satallax [15]. As it can be seen in Table 2, all proof tasks were solved successfully, which provides strong evidence in addition to theoretical results above, for the soundness (and completeness) of our embedding. In most cases, the desired properties could be automatically proved in less than $10ms$. This also provides supporting practical evidence for our approach.

6 Conclusion

Various techniques to automate reasoning in many-valued logics have been presented in the literature [21, 3]. The approach presented here, which employs a semantic embedding in classical higher-order logic, provides a theoretically and pragmatically appealing alternative. In particular, it is readily applicable

```
%-- Truth degrees
thf(n_type,type,( n: ($o>$o)>$o )).
thf(n_def,definition,( n=(^[X:$o>$o]:$false) )).
thf(nn_type,type,( nn: ($o>$o)>$o )).
thf(nn_def,definition,( nn = (^[X:$o>$o]:(~(X@$false)&~(X@$true))) )).
...
thf(ftb_type,type,( ftb: ($o>$o)>$o )).
thf(ftb_def,definition,(ftb = (^[X:$o>$o]:((X@$false)|(X@$true))) )).
thf(all_type,type,( all: ($o>$o)>$o )).
thf(all_def,definition,( all = (^[X:$o>$o]:$true) )).
%-- Truthful/Truthless subsets
thf(tpos_subset_type,type,( tpos_subset: ((($o>$o)>$o)>($o>$o)>$o) )).
thf(tpos_subset_def,definition,( tpos_subset =
 (^[T:($o>$o)>$o,X:$o>$o]:((T@X)&(X@$true))) )).
thf(tneg_subset_type,type,( tneg_subset: ((($o>$o)>$o)>($o>$o)>$o) )).
thf(tneg_subset_def,definition,( tneg_subset =
 (^[T:($o>$o)>$o,X:$o>$o]:((T@X)&~(X@$true))) )).
...
%-- Orderings
thf(ord_t_type,type,( ord_t: ((($o>$o)>$o)>(($o>$o)>$o)>$o) )).
thf(ord_t_def,definition,( ord_t = (^[X:($o>$o)>$o,Y:($o>$o)>$o]:
 (![A:$o>$o]: (((tpos_subset@X@A)=>(tpos_subset@Y@A))
            &((tneg_subset@Y@A)=>(tneg_subset@X@A))))) )).
...
%-- Lattice operations
thf(inverse_t_type,type,( inverse_t: ((($o>$o)>$o)>($o>$o)>$o) )).
thf(inverse_t_def,definition,( inverse_t = (^[T:($o>$o)>$o,X:$o>$o]:
 (T@(^[Y:$o]:((~(Y)=>(X@$false))&(Y=>~(X@$true)))))) )).
thf(join_t_ty,type,( join_t: (($o>$o)>$o)>(($o>$o)>$o)>($o>$o)>$o )).
thf(join_t_def,definition,( join_t = (^[X:($o>$o)>$o,Y:($o>$o)>$o]:
 (union@(union@(tpos_subset@X)@(tpos_subset@Y))
      @(intersect@(tneg_subset@X)@(tneg_subset@Y)))) )).
thf(meet_t_ty,type,( meet_t: (($o>$o)>$o)>(($o>$o)>$o)>($o>$o)>$o )).
thf(meet_t_def,definition,( meet_t = (^[X:($o>$o)>$o,Y:($o>$o)>$o]:
(union@(union@(tneg_subset@X)@(tneg_subset@Y))
     @(intersect@(tpos_subset@X)@(tpos_subset@Y)))) )).
...
```

Figure 1: THF encoding excerpt. Some truth-degrees and operations are omitted for brevity. Some notes concerning the THF format: The type of truth-values is written `$o`, and `$o>$o` represents the type of (characteristic) functions from truth-values to truth-values, etc. `$true` and `$false` represent truth and falsity. λ-abstractions and applications are denoted with `^` and `@`, respectively. `~`, `|`, `&`, `=>` encode negation, disjunction, conjunction and implication, and `!` denotes universal quantification. Comments are lines starting with `%`.

From	Statement	Result	Time
Prop 3.2 1.	$\forall s,t.(\mathbf{T} \in s \wedge \mathbf{T} \in t) \Leftrightarrow \mathbf{T} \in s \sqcap_t t$	Theorem	$8ms$
	$\forall s,t.(\mathbf{B} \in s \wedge \mathbf{B} \in t) \Leftrightarrow \mathbf{B} \in s \sqcap_t t$	Theorem	$9ms$
	$\forall s,t.(\mathbf{F} \in s \vee \mathbf{F} \in t) \Leftrightarrow \mathbf{F} \in s \sqcap_t t$	Theorem	$8ms$
	$\forall s,t.(\mathbf{N} \in s \vee \mathbf{N} \in t) \Leftrightarrow \mathbf{N} \in s \sqcap_t t$	Theorem	$9ms$
Prop 3.2 2.	$\forall s,t.(\mathbf{T} \in s \vee \mathbf{T} \in t) \Leftrightarrow \mathbf{T} \in s \sqcup_t t$	Theorem	$8ms$
	$\forall s,t.(\mathbf{B} \in s \vee \mathbf{B} \in t) \Leftrightarrow \mathbf{B} \in s \sqcup_t t$	Theorem	$8ms$
	$\forall s,t.(\mathbf{F} \in s \wedge \mathbf{F} \in t) \Leftrightarrow \mathbf{F} \in s \sqcup_t t$	Theorem	$8ms$
	$\forall s,t.(\mathbf{N} \in s \wedge \mathbf{N} \in t) \Leftrightarrow \mathbf{N} \in s \sqcup_t t$	Theorem	$9ms$
Prop 3.2 3.	$\forall s,t.(\mathbf{T} \in s \wedge \mathbf{T} \in t) \Leftrightarrow \mathbf{T} \in s \sqcup_f t$	Theorem	$8ms$
	$\forall s,t.(\mathbf{N} \in s \wedge \mathbf{N} \in t) \Leftrightarrow \mathbf{N} \in s \sqcup_f t$	Theorem	$8ms$
	$\forall s,t.(\mathbf{F} \in s \vee \mathbf{F} \in t) \Leftrightarrow \mathbf{F} \in s \sqcup_f t$	Theorem	$8ms$
	$\forall s,t.(\mathbf{B} \in s \vee \mathbf{B} \in t) \Leftrightarrow \mathbf{B} \in s \sqcup_f t$	Theorem	$8ms$
Prop 3.2 4.	$\forall s,t.(\mathbf{T} \in s \vee \mathbf{T} \in t) \Leftrightarrow \mathbf{T} \in s \sqcap_f t$	Theorem	$8ms$
	$\forall s,t.(\mathbf{N} \in s \vee \mathbf{N} \in t) \Leftrightarrow \mathbf{N} \in s \sqcap_f t$	Theorem	$8ms$
	$\forall s,t.(\mathbf{F} \in s \wedge \mathbf{F} \in t) \Leftrightarrow \mathbf{F} \in s \sqcap_f t$	Theorem	$8ms$
	$\forall s,t.(\mathbf{B} \in s \wedge \mathbf{B} \in t) \Leftrightarrow \mathbf{B} \in s \sqcap_f t$	Theorem	$8ms$
Def 3.6 1.	$\forall a,b.a \leq_t b \Rightarrow -_t b \leq_t -_t a$	Theorem	$421ms$
	$\forall a,b.a \leq_f b \Rightarrow -_t a \leq_f -_t b$	Theorem	$422ms$
	$\forall a,b.a \leq_i b \Rightarrow -_t a \leq_i -_t b$	Theorem	$8ms$
	$\forall a. -_t -_t a = a$	Theorem	$15ms$
Def 3.6 2.	$\forall a,b.a \leq_t b \Rightarrow -_f a \leq_t -_f b$	Theorem	$419ms$
	$\forall a,b.a \leq_f b \Rightarrow -_f b \leq_f -_f a$	Theorem	$423ms$
	$\forall a,b.a \leq_i b \Rightarrow -_f a \leq_i -_f b$	Theorem	$9ms$
	$\forall a. -_f -_f a = a$	Theorem	$17ms$

Table 2: Automated verification results of soundness relevant properties. The time results refer to the measurements with Satallax 2.7.

(with off the shelf higher-order reasoners), it enables object-level and meta-level reasoning and it supports further logic extensions and combinations.

Various extensions of many-valued logics have been studied in the literature. Examples include many-valued modal logics or many-valued predicate logics.

Respective extensions of our embedding of *SIXTEEN* in HOL are analogously feasible. In particular, it should be possible to adapt the embedding of quantified modal logics [12] and combine it with the work presented here. Shramko and Wansing [26, pp.216], for example, present an idea to develop first-order trilattice logics from modal trilattice logics. In this context, a Kripke-style semantics for quantification is provided in the following form:

$$M, \alpha \models \forall x A \text{ iff for every state } \beta: \text{ if } \alpha R_x \beta, \text{ then } M, \beta \models A$$

In previous work [12] we have illustrated that similar clauses (e.g. the modal box operator) can easily be encoded. New is here that the accessibility relation R_x depends on the individual x, but such a dependency can easily be captured.

Further future work includes the application of the presented automation technique to more practically motivated examples. We are positive that this approach can indeed be used to deal with meaningful reasoning tasks where e.g. linguistic vagueness or uncertainty is involved. Also the meta-level reasoning capabilities of our approach leave room for much further work. In fact, we are positive that many meta-level statements and theorems in textbooks and publications can at least partially be verified (or falsified) with it.

Moreover, it should be possible to provide human-intuitive proof tactics in proof assistants to support interactive proof development. It was shown in previous work that similar tactics for modal logic could successfully be employed in such proof assistants [**?**]. In combination with proof automation, this should lead to fruitful employment for computer-aided argumentation and reasoning within theoretical philosophy.

Acknowledgment

This work has been supported by the German National Research Foundation (DFG) under grants BE 2501/11-1 (Leo-III) and BE2501/9-2 (Ontoleo).

References

[1] Peter B. Andrews. General models and extensionality. *Journal of Symbolic Logic*, 37(2):395–397, 1972.

[2] Peter B. Andrews. General models, descriptions, and choice in type theory. *Journal of Symbolic Logic*, 37(2):385–394, 1972.

[3] M. Baaz, C. Fermller, and G. Salzer. *Handbook of Automated Reasoning*, chapter Automated Deduction in Many-Valued Logics. Elsevier, 2001.

[4] H. P. Barendregt, W. Dekkers, and R. Statman. *Lambda Calculus with Types*. Perspectives in logic. Cambridge University Press, 2013.

[5] N. D. Belnap. A useful four-valued logic. In J. M. Dunn and G. Epstein, editors, *Modern Uses of Multiple-Valued Logic*, volume 2 of *Episteme*, pages 5–37. Springer NL, 1977.

[6] N. D. Belnap. How a computer should think. In A. Anderson, N. D. Belnap, and J. M. Dunn, editors, *Entailment: The Logic of Relevance and Necessity*, volume 2. Princeton University Press, 1992.

[7] C. Benzmüller. Gödel's ontological argument revisited – findings from a computer-supported analysis (invited). In Ricardo Souza Silvestre and Jean-Yves Béziau, editors, *Handbook of the 1st World Congress on Logic and Religion, João Pessoa, Brazil*, page 13, 2015.

[8] C. Benzmüller. Invited Talk: On a (Quite) Universal Theorem Proving Approach and Its Application in Metaphysics. In Hans De Nivelle, editor, *TABLEAUX 2015*, volume 9323 of *LNAI*, pages 209–216, Wroclaw, Poland, 2015. Springer.

[9] C. Benzmüller, Theiss F., L. Paulson, and A. Fietzke. LEO-II – A Cooperative Automatic Theorem Prover for Higher-Order Logic (System Description). In *Automated Reasoning, IJCAR 2008, Sydney, Australia, Proceedings*, volume 5195 of *LNCS*, pages 162–170. Springer, 2008.

[10] C. Benzmüller, D. Gabbay, V. Genovese, and D. Rispoli. Embedding and automating conditional logics in classical higher-order logic. *Annals of Mathematics and Artificial Intelligence*, 66(1-4):257–271, 2012.

[11] C. Benzmüller and D. Miller. Automation of Higher-Order Logic. In Dov M. Gabbay, Jörg H. Siekmann, and John Woods, editors, *Handbook of the History of Logic, Volume 9 — Computational Logic*, pages 215–254. North Holland, Elsevier, 2014.

[12] C. Benzmüller and L. Paulson. Quantified multimodal logics in simple type theory. *Logica Universalis (Special Issue on Multimodal Logics)*, 7(1):7–20, 2013.

[13] C. Benzmüller, L. Weber, and B. 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*, pages 53–54, 2015.

[14] C. Benzmüller and B. Woltzenlogel Paleo. Higher-order modal logics: Automation and applications. In Adrian Paschke and Wolfgang Faber, editors, *Reasoning Web 2015*, number 9203 in LNCS, pages 32–74, Berlin, Germany, 2015. Springer.

[15] C. E. Brown. Satallax: An Automatic Higher-Order Prover. In *Automated Reasoning - 6th International Joint Conference, IJCAR 2012, Manchester, UK, June 26-29, 2012. Proceedings*, pages 111–117, 2012.

[16] A. Church. A Set of Postulates for the Foundation of Logic. *Annals of Mathematics*, 33(2):346–366, 1932.

[17] A. Church. A Set of Postulates for the Foundation of Logic, Second Paper. *The Annals of Mathematics*, 34(4):839–864, October 1933.

[18] A. Church. A formulation of the simple theory of types. *J. Symb. Log.*, 5(2), 1940.

[19] K. Gödel. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme. *Monatshefte für Mathematik und Physik*, 38(1):173–198, 1931.

[20] L. Henkin. Completeness in the theory of types. *Journal of Symbolic Logic*, 15(2):81–91, 06 1950.

[21] R. Hhnle. *Automated deduction in multiple-valued logics*. Clarendon Press, Oxford, 1993.

[22] J. Łukasiewicz. O logice trojwartosciowej. *Ruch Filozoficzny*, 5:170–171, 1920.

[23] T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer, 2002.

[24] S. P. Odintsov. On Axiomatizing Shramko-Wansings Logic. *Studia Logica*, 91(3):407–428, 2009.

[25] Y. Shramko and H. Wansing. Some Useful 16-Valued Logics: How a Computer Network Should Think. *Journal of Philosophical Logic*, 34(2):pp. 121–153, 2005.

[26] Y. Shramko and H. Wansing. *Truth and Falsehood: An Inquiry into Generalized Logical Values*. Trends in Logic. Springer Netherlands, 2011.

[27] A. Steen and M. Wisniewski. Embedding of First-Order Nominal Logic into HOL. In Jean-Yves Beziau, Safak Ural, Arthur Buchsbaum, Iskender Tasdelen, and Vedat Kamer, editors, *5th World Congress and School on Universal Logic (UNILOG'15)*, pages 337–339, Istanbul, Turkey, 2015.

[28] G. Sutcliffe. The TPTP Problem Library and Associated Infrastructure: The FOF and CNF Parts, v3.5.0. *Journal of Automated Reasoning*, 43(4):337–362, 2009.

[29] G. Sutcliffe and C. Benzmüller. Automated Reasoning in Higher-Order Logic using the TPTP THF Infrastructure. *Journal of Formalized Reasoning*, 3(1):1–27, 2010.

Alexander Steen
Freie Universität Berlin
Institute of Computer Science
a.steen@fu-berlin.de

Christoph Benzmüller
Stanford University
CSLI/Cordula Hall
c.benzmueller@gmail.com