Masterarbeit am Institut für Informatik der Freien Universität Berlin

Arbeitsgruppe Dahlem Center for Machine Learning and Robotics

A Framework for Higher-Order Modal Logic Theorem Proving

Tobias Gleißner
Matrikelnummer: 4625710
tobias.gleissner@fu-berlin.de

Erstgutachter: Alexander Steen
Zweitgutachter: Christoph Benzmüller

Berlin, August 28, 2019

Abstract

Modal logic is a formalism suitable for modelling problems in artificial intelligence, computer science, philosophy and many other disciplines. Its automation in all its flavours can easily be achieved employing the semantic embedding approach and reusing existing reasoning software: This thesis presents a procedure that encodes a problem of higher-order modal logic in higher-order logic which is implemented in the Modal Embedding Tool for the input language TPTP THF which was extended for this purpose. By combining this software with one or more compliant reasoners a competitive system is created that can handle countless more variants of modal logic than any other theorem prover available to date.

Keywords

Higher-Order Modal Logic, Quantified Modal Logic, Semantic Embedding, Proof Automation

Eidesstattliche Erklärung

Ich versichere hiermit an Eides Statt, dass diese Arbeit von niemand anderem als meiner Person verfasst worden ist. Alle verwendeten Hilfsmittel wie Berichte, Bücher, Internetseiten oder ähnliches sind im Literaturverzeichnis angegeben, Zitate aus fremden Arbeiten sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungskommission vorgelegt und auch nicht veröffentlicht.

August 28, 2019

Tobias Gleißner

Contents

1 Introduction

1.1 Context

Sound argumentation is the most important concern when creating new knowledge or debating social, political and philosophical issues. It governs scientific inquiry, questions of morality and our everyday-life. Usually, an argument is presented in a mostly narrative manner leaving it ambiguous and susceptible to other people and also often to oneself, while at the same time the argumentation's steps themselves may not become clear and are prone to error.
Overcoming these problems is a major task in an environment that has a large or partly contradictory set of rules and norms resulting not only but also from unclear ethical considerations, a vastly growing amount of information, and, the fast increasing quantity of new (scientific) discoveries. Therefore keeping track and verifying the validity of a given discourse in any domain becomes almost impossible.
Employing formal representations may provide a possible angle on one or more of those problems. Usually, such formal representation is achieved with a set of propositions (*terms*) that bear a very specific meaning (*semantics*) which hopefully resolves ambiguities when modelling the object(s) of inquiry. Additionally such a system possesses a set of rules (*calculus*) that allows the derivation (*proof*) of facts with respect to the system. This leaves the debate "merely" with the challenges of reasonably modelling the domain, thereby resolving the ambiguities, and the choice of the premises. The rest of the argument involves a purely mechanical procedure that, if carried out with rigor, will lead to the right answer, given the premises and correct formalization of the domain. Of course ambiguity at the same time allows for more flexible and compact representation of information, but a process of concise formalization may discover that important details have been left out in an argument or assumptions are missing because they have been considered implicit knowledge.
The manual derivation of an answer or checking its validity can prove very difficult. There are several reasons for this: First of all, humans, when carrying out a formal proof use abstractions on those systems (as employed in mathematical contexts) that can be vulnerable to subtle mistakes. Second, the sheer complexity of such a derivation may lead to a solution so large and confusing that it is hard to verify even for a group of experts as it is the case in the proof of the Kepler's Conjecture [30].
Since the mechanical nature of many calculi destines them for automation, it caused the development of computer-assisted reasoning tools such as automated theorem provers (ATP). An ATP can be supplied with a set of premises and a hypothesis as input (of a specific logic) in a machine-readable format like TPTP [51]. Next, it automatically tries creating a derivation for the hypothesis in question. The success of this process is highly dependent on the input, the heuristics of the ATP, the available computational power and most importantly, the computational complexity of the logic itself. The most versatile logics that are usually considered lie in the realm of semi-decidability, i.e. there exists an algorithm, that eventually creates any true statement of such a logic, but without runtime prediction of the procedure, and, there does not exist an algorithm, that will disprove any

false statement. However, ATPs have been successfully employed in various scenarios [30, 33, 14, 11, 55] and can be expected to become even more powerful in the future.

1.2 Presentation of the Problem

Non-classical logics possess built-in mechanisms for special purposes that would be complex to model in the popular classical logics and probably result in incomprehensible logic formulas. Special purposes could comprise characterizing time, knowledge, believe, undefinedness, agents, different kinds of implications, and many more. In this thesis, the non-classical logic of interest is higher-order modal logic (HOML) which is useful in various domains (see sect. 1.3). Only few systems exist for reasoning in modal logics, especially in quantified modal logics (see sect. 1.4) and all of them support only some fragment of it. Unfortunately, reasoning systems require a sophisticated (theoretical) design and are time-consuming to implement. Hence building new provers is not a viable option for automating (all) non-classical logics of interest with their vast amount of semantics choices, especially in the case of HOML.
The *semantic embedding* approach remedies this situation. The idea is to encode a problem of the source logic (HOML in this case) in a general purpose target logic which in this case is classical higher-order logic (HOL), for which automated reasoning tools already exist. Software implementing this approach can be developed comparatively quickly and easily, thereby eliminating the immediate need for building special purpose provers by applying one or more of the off-the-shelf HOL ATPs [48, 16, 53, 35, 15] to an encoded problem.

1.3 Relevance

Modalities are expressions that ”qualify the truth of a judgment” [25]. Examples for such an alteration are topic of deontic logics that talk about permissions and obligations, temporal logics that capture states unfolding in a timeline, or conditional logics, that focus on the meaning of the concept of implication. Worth mentioning are also logics for verification of computer software e.g. linear temporal logic and logics for alethic or doxastic contexts. Many manifestations of these logics are actually modal logics and can share the same semantic base.
The automation of those logics not only helps in identifying and validating propositions in these fields but can also be of use in the verification of derivations in publications as it was performed in the computer-assisted analysis of the Anderson-Hájek controversy [11]. More interestingly, when modelling the domain of inquiry, computer-assisted reasoning can easily reveal fine differences in the meaning of similar statements as it happened for a renewed version of Gödel’s ontological argument by Scott [46, 14]. Automated reasoning software can assist in exposing mistakes in the modelling of formalized objects by examining desired properties of these objects under slightly changed semantics for the underlying logic. This holds especially when pursuing the semantic embedding approach. It simplifies the implementation of an automation software on such a magnitude that it becomes possible to explore all subtleties with comparably little effort.

1.4 Related Work

Among the prominent reasoners for modal logic are MleanSeP, MLeanTaP, MleanCoP [41] and the combination f2p-MSPASS [32] for first-order modal logic. Also there exists a large variety of propositional reasoners.[1] Previous work [9] realizes the semantic embedding approach only for first-order-modal logic and some of its variants. The software and shallow semantic embedding for higher-order modal logic developed in this thesis are an expansion of the Modal Embedding Tool (MET) and its theory [26, 28, 27] which is based on the work of Benzmüller et al. [12] The latter has its roots in the standard translation of (first-order) modal logic [37] that performs a similar transformation from a semantic perspective (of modal logic) but does not explicitly express the meta-logical parts of the source logic as it is the case in a shallow semantic embedding. Shallow semantic embeddings are also available for other non-classical logics such as intuitionistic logics [6], free logics [10], hybrid logics [56], many-valued logics [47], access control logics [2], input-output logics [21], and dyadic deontic logics [4].

1.5 Objectives of this Thesis

This thesis is intended to be the start of a framework for modal logics. First, it should clarify the syntax and semantics of higher-order modal logics and its variants by providing clean and comprehensible definitions. Furthermore, a suitable machine-readable input language for HOML has to be devised, based on the TPTP language THF [8] which can be conservatively extended for describing terms and formulas of modal logics and specifying the semantics of the logic. The collection and curation of HOML problems and their representation in this format present another objective of this thesis. Subsequently, the assessment of the MET's performance should play an important part since a positive outcome would justify the semantic embedding approach. Finally, the further development of the MET is focus of this thesis. This comprises not only the support of more semantics choices but also the implementation and evaluation of different encodings for identical semantics variants. Moreover, the encoding underlying the MET will be presented in detail.

1.6 Structure of this Thesis

This thesis is structured as follows: Sect. 2 presents higher-order modal logics and its variants. It provides details on syntax and semantics of a general version of modal logics that is later restricted accordingly for each variation. Encodings that apply the shallow semantic embedding approach to modal logic (and its variants) are described in sect. 3 for target logic HOL. A brief overview of the semantic embedding's implementation and the input syntax is shown in sect. 4. Subsequently, the performance of the MET paired with higher-order reasoning systems is depicted in sect. 5 and measured against the most advanced native reasoner MleanCoP. Finally, the thesis concludes with a summary and future work in sect. 6.

[1]An extensive list of modal reasoning software is maintained by Schmidt [43].

2 Higher-Order Modal Logic

Higher-order modal logic (HOML) is an extension of classical Higher-Order Logic (HOL) that allows for qualifying "the truth of a judgment" [25]. It appeared as a formalism for discussing modalities like *necessarily* or *possibly* at first and was reused in contexts of obligations, permissions, beliefs and time. The signature of classical (propositional) logic was extended with the operators $\Box$ and $\Diamond$ that should capture the notions of *necessity* and *possibility*, respectively. Therefore the inference rule $\Phi \vdash \Box\Phi$, called *rule of necessitation* and the distribution axiom $\Box(\Phi \supset \Psi) \supset (\Box\Phi \supset \Box\Psi)$ or *schema K* were added to the calculus of the original logic.
Modal logic historically constitutes a logic containing an operator $\Box$ that satisfies at least schema K and is also called *normal modal logics*. The interesting difference to the connectives of classical logics is the lack of truth functionality of these operators i.e. the truth of $\Box\Phi$ does not depend on the truth-value of the argument Φ exclusively. As a consequence $\Phi \supset \Box\Phi$ is not a theorem in modal logic. Today this proof-theoretic description of modal logics is replaced by a semantic characterization called *Kripke semantics* or *possible worlds semantics*.
This chapter introduces the syntax and large variety of choices for the (Kripke) semantics of modal logic with higher-order quantification. Most definitions and notations are borrowed from Gleißner et al. [28] which has its roots in Benzmüller et al. [13], Muskens [39] and Wiesniewski et al. [57].

2.1 Syntax and Semantics

In this thesis HOML is considered a logic based on typed λ-calculus [1]. It provides the construction of functions via λ-abstractions which again can be applied to appropriate terms (function application). Functions of arbitrary arity are realized through *currying* [24, 45] i.e. λ-abstractions can emit functions as values. The underlying typed λ-calculus features the usual semantics [1], in this case α-, β- and η-conversion are assumed and all terms are considered β- and η-normalized. [2] [3]

Syntax. All terms are annotated with types that put restrictions on the formation of terms in order to overcome problems like Russel's Paradox [54].

Definition 1 (Types)
The *set of types* [4] $\mathcal{T}$ is the smallest set that contains the following elements:

o	as Boolean type,
ι	as type for the domain of individuals, and
$\tau\nu$	the function types with $\tau, \nu \in \mathcal{T}$.

⌟

[2]For details on the typed λ-calculus see *Barendregt - Lambda Calculus with Types* [1].
[3]Different semantics of the typed λ-calculus are depicted in *Benzmüller, Brown and Kohlhase - Higher-Order Semantics and Extensionality*[3].
[4]HOML may also be defined introducing further base types, which is omitted here for simplicity.

Types that are not function types are called *base types.*
The terms of HOML are constructed as follows:

Definition 2 (Terms)
Let $i \in I$ where I is a non-empty finite set, X_τ is a *variable* from a countably infinite set $\mathcal{V}_\tau$ of variable symbols for each type $\tau \in \mathcal{T}$ and $c_\tau \in \Sigma_\tau$ is a symbol for a constant from the signature $\Sigma \equiv \bigcup_{\tau \in \mathcal{T}} \Sigma_\tau$, that contains at least the primitive logical connectives for negation, disjunction and, for each type, both equality and universal quantification. Hence for all $\tau \in \mathcal{T}$ $\{\neg_{oo}, \vee_{ooo}, \Pi^\tau_{(\tau o)o}\} \subseteq \Sigma$. Let $\tau, \nu \in \mathcal{T}$. The *terms of HOML* are generated by the following grammar:

$$s, t \quad ::= \quad c_\tau \mid X_\tau \mid (\lambda X_\tau.\, s_\nu)_{\tau\nu} \mid (s_{\tau\nu}\, t_\tau)_\nu \mid (\Box^i_{oo}\, s_o)_o$$

⌟

Terms of type o are referred to as *formulas.* I, $\mathcal{V}_\tau$, Σ and Σ_τ are used as defined above in the remainder.

Semantics. A model in HOML is constituted by a multi-modal frame, domain collections and interpretations. The multi-modal frame consists of a set of possible worlds and binary relations on those worlds. The available mathematical objects are defined through the domain collections and the meaning of constant symbols are determined by there interpretation. There is one domain collection and one interpretation for every single world leading towards a different denotation of HOML terms on distinct worlds when talking about the value of a term. This formulas are true on some world but not on the other.

Definition 3 (Multi-modal frame)
A *multi-modal frame* is a tuple $(\mathcal{W}, \{\mathcal{R}^i\}_{i \in I})$ where $\mathcal{W}$ is a non-empty set of arbitrary cardinality and $\{\mathcal{R}^i\}_{i \in I}$ is a family of binary relations on $\mathcal{W}$. Elements of $\mathcal{W}$ are called *worlds* or *possible worlds.* Elements of $\{\mathcal{R}^i\}_{i \in I}$ are called *accessibility relations.* ⌟

Note that this is a more generalized definition of Kripke structure or Kripke frame as found in the original paper by Kripke [36] since it offers more than one modality ($\Box$-operator). [5]

Definition 4 (Domain collection)
A *domain collection* $\{D_\tau\}_{\tau \in \mathcal{T}}$ is a family of non-empty sets D_τ, such that

1. D_o is the domain of Booleans with $D_o \equiv \{\top, \bot\}$ where $\top$ and $\bot$ represent truth and falsehood,
2. D_ι is the domain of individuals, and
3. $D_{\tau\nu}$ is the domain of total functions that map elements of D_τ to elements of D_ν for $\tau, \nu \in \mathcal{T}$.

⌟

[5] A more generalized version might involve more than one set of worlds permitting accessibility relations being defined on different sets of worlds.

In classical higher-order logic the domain collection is usually referred to as "frame". Since modal logic employs this term for the tuple of worlds and their relations it is not used here in order to avoid ambiguity.

Definition 5 (Interpretation)
Let $\{D_\tau\}_{\tau\in\mathcal{T}}$ be a domain collection. An *interpretation* $\mathcal{I}$ is a function that maps every constant symbol $c_\tau \in \Sigma_\tau$ to an element $d \in D_\tau \in \{D_\tau\}_{\tau\in\mathcal{T}}$ for any type $\tau \in \mathcal{T}$. The interpretation of the primitive logical connectives and the universal quantification(s) are always defined as:

$$\mathcal{I}(\neg_{oo}) \equiv f : D_o \to D_o \text{ with } y \overset{f}{\mapsto} \begin{cases} \bot & \text{if } y \equiv \top \\ \top & \text{otherwise} \end{cases}$$

$$\mathcal{I}(\vee_{ooo}) \equiv f : D_o \times D_o \to D_o \text{ with } (y,z) \overset{f}{\mapsto} \begin{cases} \bot & \text{if } y \equiv \bot \text{ and } z \equiv \bot \\ \top & \text{otherwise} \end{cases}$$

$$\mathcal{I}(\Pi^\tau_{(\tau o)o}) \equiv f : D_{\tau o} \to D_o \text{ with } y \overset{f}{\mapsto} \begin{cases} \top & \text{if } y(d) \equiv \top \text{ for all } d \in D_\tau \\ \bot & \text{otherwise} \end{cases}$$

⌟

The remaining logical connectives and quantifiers for conjunction, implication, existential quantification and the $\Diamond$-operator can be, respectively, defined as usual:

$$\begin{aligned} \wedge_{ooo} &\equiv \lambda s_o.\, \lambda t_o.\, \neg(\neg s \vee \neg t) \\ \supset_{ooo} &\equiv \lambda s_o.\, \lambda t_o.\, (\neg s) \vee t \\ \Sigma^\tau_{(\tau o)o} &\equiv \lambda s_{\tau o}.\, \neg\Pi^\tau_{(\tau o)o} s \\ \Diamond^i_{oo} &\equiv \lambda s_o.\, \neg\big(\Box^i(\neg s)\big) \end{aligned}$$

Definition 6 (Standard model)
A *standard model* for HOML is a tuple $(\mathcal{W}, \{\mathcal{R}^i\}_{i\in I}, \{\mathcal{D}_w\}_{w\in\mathcal{W}}, \{\mathcal{I}_w\}_{w\in\mathcal{W}})$ where

1. $(\mathcal{W}, \{\mathcal{R}^i\}_{i\in I})$ comprises a multi-modal frame,
2. $\{\mathcal{D}_w\}_{w\in\mathcal{W}}$ is a family of domain collections, one domain collection $\mathcal{D}_w \equiv \{D^w_\tau\}_{\tau\in\mathcal{T}}$ for each world $w \in \mathcal{W}$ with $D^w_{\tau\nu} \equiv D^{w D^w_\tau}_\nu$ for $\tau, \nu \in \mathcal{T}$, and
3. $\{\mathcal{I}_w\}_{w\in\mathcal{W}}$ is a family of interpretations, one for each world $w \in \mathcal{W}$.

⌟

Preparatory to the valuation function that assigns a meaning to all terms of HOML the *variable assignment* is introduced as a function, that substitutes variables with actual objects of the corresponding domain.

Definition 7 (Variable assignment)
Let $\{D_\tau\}_{\tau\in\mathcal{T}}$ be a domain collection. A *variable assignment* g maps variables $X_\tau \in \mathcal{V}_\tau$ to elements of $D_\tau \in \{D_\tau\}_{\tau\in\mathcal{T}}$ for any type $\tau \in \mathcal{T}$.
A *substitution of variable* X *in* g[6], written $g[d_\tau/X_\tau]$, denotes the variable assignment g' that is identical to g except that in g' the variable X_τ is mapped to d_τ, i.e. $g'(X_\tau) \equiv d_\tau$ and for all $\tau \in \mathcal{T}$ and $Y_\tau \in \mathcal{V}_\tau$ it holds that $g'(Y_\tau) \equiv g(Y_\tau)$ if $Y_\tau \not\equiv X_\tau$.
For a family of variable assignments $g \equiv \{g_w\}_{w\in\mathcal{W}}$ the substitution of variable X on exactly one variable assignment g_w of g that leaves all elements of g the same as before except for g_w which is altered as described aboveis denoted by $g[d_\tau/X_\tau]_w$. ⌟

Now the *value* of a HOML term can be determined by the *valuation function*[7]:

Definition 8 (Valuation function)
The *value* $\|u_\tau\|^{M,g,w}$ of a term u_τ of type $\tau \in \mathcal{T}$ with respect to a model $M \equiv (\mathcal{W}, \{\mathcal{R}^i\}_{i\in I}, \{\mathcal{D}_w\}_{w\in\mathcal{W}}, \{\mathcal{I}_w\}_{w\in\mathcal{W}})$ under a family of variable assignments $g \equiv \{g_w\}_{w\in\mathcal{W}}$ at world $w \in \mathcal{W}$ is an element of $D^w_\tau \in \mathcal{D}_w$, defined by

$$\begin{aligned}
\|X_\tau\|^{M,g,w} &\equiv g_w(X_\tau)\\
\|c_\tau\|^{M,g,w} &\equiv \mathcal{I}_w(c_\tau)\\
\|\,(\lambda X_\tau.\, s_\nu)_{\tau\nu}\,\|^{M,g,w} &\equiv f : D^w_\tau \to D^w_\nu \text{ with } y \overset{f}{\mapsto} \|s_\nu\|^{M,g[y/X_\tau]_w,w}\\
\|\,(s_{\tau\nu}\, t_\tau)_\nu\,\|^{M,g,w} &\equiv \|s_{\tau\nu}\|^{M,g,w}\big(\|t_\tau\|^{M,g,w}\big)\\
\|\Box^i_{oo}\, s_o\|^{M,g,w} &\equiv \begin{cases} \top & \text{if } \|s_o\|^{M,g,v} \equiv \top \text{ for all } v \in \mathcal{W} \text{ s.t. } (w,v) \in \mathcal{R}^i\\ \bot & \text{otherwise} \end{cases}
\end{aligned}$$
⌟

As a result of Gödel's Incompleteness Theorem, complete mechanization of HOML with standard semantics cannot be achieved. Instead, an adaption of Henkin semantics [31] is introduced and Henkin models are assumed for the remainder of this thesis.

Definition 9 (Henkin model)
A *Henkin model* for HOML is a tuple $(\mathcal{W}, \{\mathcal{R}^i\}_{i\in I}, \{\mathcal{D}_w\}_{w\in\mathcal{W}}, \{\mathcal{I}_w\}_{w\in\mathcal{W}})$ where

1. $(\mathcal{W}, \{\mathcal{R}^i\}_{i\in I})$ comprises a multi-modal frame,
2. $\{\mathcal{D}_w\}_{w\in\mathcal{W}}$ is a family of domain collections, one domain collection $\mathcal{D}_w \equiv \{D^w_\tau\}_{\tau\in\mathcal{T}}$ for each world $w \in \mathcal{W}$ with $D^w_{\tau\nu} \subseteq {D^w_\nu}^{D^w_\tau}$ for $\tau, \nu \in \mathcal{T}$,
3. $\{\mathcal{I}_w\}_{w\in\mathcal{W}}$ is a family of interpretations, one for each world $w \in \mathcal{W}$, and
4. the valuation function $\|\cdot\|$ remains a total function.

⌟

[6] Sometimes also called *X-variant of g*.
[7] Note that the $\Box$-operator cannot be defined through interpretation since its semantics requires access on the model level.

Henkin models do not demand the full set of functions for each domain $D^w_{\tau\nu}$. Since the valuation function is required to be a total function there is a denotation for all terms of HOML. This leads to a more general model concept, especially every standard model is also a Henkin model. As a consequence all theorems in *Henkin semantics* (all Henkin models are considered) are theorems in *standard semantics* (only standard models are considered), but not vice versa. For the remainder of this thesis the term Model in the context of modal logics refers to a Henkin model.
Finally, truth and validity relations can be defined.

Definition 10 (Truth and validity)

$M, g, w \vDash^{HOML} s_o$	A term s_o is *true* in a model M for a world w under an assignment g if and only if $\|s_o\|^{M,g,w} \equiv \top$.
$M, w \vDash^{HOML} s_o$	A term s_o is *locally valid* in a model M for a world w if and only if $M, g, w \vDash^{HOML} s_o$ for all assignments g.
$M \vDash^{HOML} s_o$	A term s_o is *globally valid* in a model M if and only if $M, w \vDash^{HOML} s_o$ for all worlds w.
$\vDash^{HOML} s_o$	A term s_o is *valid* if and only if $M \vDash^{HOML} s_o$ for all models M.

$\lrcorner$

These definitions of $\vDash^{HOML}$ can also be used on a set of formulas in place of s_o in which the relations have to hold on all elements of this set.
Notions of logical consequence are defined in section 2.2.4.

2.2 Semantics Variations

This fairly general and refined version of modal logic can be supplemented with a variety of semantic options where each one of them has its particular applications. The upcoming paragraphs provide an overview of these choices for constructing logics that are tailored to specific purposes. The variants dimensions consist of

- the axiomatization of the $\Box^i$-operators,
- the semantics of quantification,
- the interpretation of constants, and
- the notion of logical consequence.

2.2.1 Modality Axiomatization

The first variation for a concrete modal logic at hand is axiomatizing $\Box^i$-operators by adding one or more of the axiom schemes in Table 1 to the proof theory. A semantic perspective on this matter offers Sahlqvist [44] who discovered that the postulation of such axioms impose certain properties on the accessibility relation of the concerned $\Box^i$-operator as listed [8] in Table 1 below:

Name	**Axiom**	**Accessibility relation property**	**Property name**
T	$\Box^i s \supset s$	$\forall w.\, r^i ww$	reflexive
B	$s \supset \Box^i \Diamond^i s$	$\forall v. \forall w.\, r^i vw \supset r^i wv$	symmetric
D	$\Box^i s \supset \Diamond^i s$	$\forall v. \exists w.\, r^i vw$	serial
4	$\Box^i s \supset \Box^i \Box^i s$	$\forall u. \forall v. \forall w.\, r^i uv \wedge r^i vw \supset r^i uw$	transitive
5	$\Diamond^i s \supset \Box^i \Diamond^i s$	$\forall u. \forall v. \forall w.\, r^i uv \wedge r^i uw \supset r^i vw$	euclidean
CD	$\Diamond^i s \supset \Box^i s$	$\forall u. \forall v. \forall w.\, r^i uv \wedge r^i uw \supset v = w$	functional
$\Box$M	$\Box^i (\Box^i s \supset \Box^i s)$	$\forall v. \forall w. \forall w.\, r^i vw \supset r^i ww$	shift-reflexive
C4	$\Box^i \Box^i s \supset \Box^i s$	$\forall u. \forall v.\, r^i uv \supset \exists w.\, r^i uw \wedge r^i wv$	dense
C	$\Diamond^i \Box^i s \supset \Box^i \Diamond^i s$	$\forall t. \forall u. \forall v.\, r^i tu \wedge r^i tv \supset \exists w.\, r^i uw \wedge r^i vw$	convergent

Table 1: Modality axiomatization and corresponding accessibility relation properties. Formulas are denoted by s, and worlds are denoted by t,u,v or w.

Often, several axioms are used in combination yielding so-called modal logic systems. An overview of the most important of such systems is displayed by Table 2:

System name	**Axiom scheme(s)**
K	K
D	K + D
T	K + T
B	K + B + T
S4	K + T + 4
S5	K + T + 5

Table 2: Popular axiom systems

Different applications employ different axiomatizations e.g. epistemic contexts the systems S4 or S5 are frequently employed and deontic problems usually require at least axiom D. For a multi-modal logic, this choice can be made for every $\Box^i$-operator independently and further bridge rules, describing the interaction between two or more $\Box^i$-operators, may be added.

[8] Axiom scheme K is omitted here since the semantics already imply its validity and it does not impose any properties on the accessibility relation.

2.2.2 Quantification

A model's domain collections may differ from world to world hence a quantifier for type $\tau \in \mathcal{T}$ may range over a different set D^w_τ on every world $w \in \mathcal{W}$. Such a type is said to yield *varying domain semantics* and this aspect of the semantics is called the *actualist* interpretation of quantification and proposes that everything there is actually exists, i.e. that there are no merely possible things [29].

Definition 11 (Varying domain)
A domain of type $\tau \in \mathcal{T}$ is called *varying* if and only if the constructed logic yields only models $M = (\mathcal{W}, \{\mathcal{R}^i\}_{i\in I}, \{\mathcal{D}_w\}_{w\in\mathcal{W}}, \{\mathcal{I}_w\}_{w\in\mathcal{W}})$, such that for all worlds $w \in \mathcal{W}$ no restrictions are imposed on $D^w_\tau \in \{\mathcal{D}_w\}_{w\in\mathcal{W}}$. ⌟

This point of view may not be adequate for all applications especially in computer science and metaphysics in which the *possibilist* interpretation of quantification play the important role. It states that domains yield so-called *constant domain semantics* i.e. any two domain collections of their corresponding type contain the same elements for any two worlds. As a consequence a quantifier for a constant domain ranges over the same set on any world.

Definition 12 (Constant domain)
A domain of type $\tau \in \mathcal{T}$ is called *constant* if and only if the constructed logic yields only models $M = (\mathcal{W}, \{\mathcal{R}^i\}_{i\in I}, \{\mathcal{D}_w\}_{w\in\mathcal{W}}, \{\mathcal{I}_w\}_{w\in\mathcal{W}})$, such that for all worlds $v, w \in \mathcal{W}$, it holds that $D^v_\tau = D^w_\tau$ for $D^w_\tau, D^v_\tau \in \{\mathcal{D}_w\}_{w\in\mathcal{W}}$. ⌟

Two further settings exist in which the *cumulative domain semantics* requires a domain of a specific type on some world to be subset of the corresponding domain in any world reachable by any accessibility relation [9]. The *decreasing domain semantics* are similar except that the subset relation is applied in reverse.

Definition 13 (Cumulative domain)
A domain of type $\tau \in \mathcal{T}$ is called *cumulative* if and only if the constructed logic yields only models $M = (\mathcal{W}, \{\mathcal{R}^i\}_{i\in I}, \{\mathcal{D}_w\}_{w\in\mathcal{W}}, \{\mathcal{I}_w\}_{w\in\mathcal{W}})$, such that for all worlds $v, w \in \mathcal{W}$, if a $\mathcal{R}^i \in \{\mathcal{R}^i\}_{i\in I}$ exists with $(v, w) \in \mathcal{R}^i$, it holds that $D^v_\tau \subseteq D^w_\tau$ for $D^w_\tau, D^v_\tau \in \{\mathcal{D}_w\}_{w\in\mathcal{W}}$. ⌟

Definition 14 (Decreasing domain)
A domain of type $\tau \in \mathcal{T}$ is called *decreasing* if and only if the constructed logic yields only models $M = (\mathcal{W}, \{\mathcal{R}^i\}_{i\in I}, \{\mathcal{D}_w\}_{w\in\mathcal{W}}, \{\mathcal{I}_w\}_{w\in\mathcal{W}})$, such that for all worlds $v, w \in \mathcal{W}$, if a $\mathcal{R}^i \in \{\mathcal{R}^i\}_{i\in I}$ exists with $(v, w) \in \mathcal{R}^i$, it holds that $D^v_\tau \supseteq D^w_\tau$ for $D^w_\tau, D^v_\tau \in \{\mathcal{D}_w\}_{w\in\mathcal{W}}$. ⌟

Finally, the quantification semantics may be chosen individually for every domain. Be aware that if the expression "varying domain semantics" comes up in the literature it usually refers to the domain of individuals only and all other domains are considered constant.

[9] In another more general setup one could think about pairs consisting of a domain and an accessibility relation that are called *cumulative* (instead of the domain itself) not imposing this structure on all other accessibility relations. Furthermore, it is unclear whether the definition used here might cause problems when introducing bridging rules.

2.2.3 Constant Denotation

In this general setting a constant $c_\tau \in \Sigma_\tau$ on some world $w \in \mathcal{W}$ is mapped by the world's interpretation function I_w to an element of the corresponding domain collection of this world. Constants with this *flexible* behavior have a world-dependent meaning i.e. a constant symbol may denote differently on different worlds.

Definition 15 (Flexible constant)
A constant $c_\tau \in \Sigma_\tau$ of the type $\tau \in \mathcal{T}$ is called *flexible* if and only if the constructed logic yields only models $M = (\mathcal{W}, \{\mathcal{R}^i\}_{i\in I}, \{\mathcal{D}_w\}_{w\in\mathcal{W}}, \{\mathcal{I}_w\}_{w\in\mathcal{W}})$, such that for all worlds $w \in \mathcal{W}$ no restrictions are imposed on the interpretation $\mathcal{I}_w(c_\tau)$ of the constant c_τ. ⌟

Constants may also be required to have the same denotation on any world. These constants are referred to as *rigid* constants.

Definition 16 (Rigid constant)
A constant $c_\tau \in \Sigma_\tau$ of the type $\tau \in \mathcal{T}$ is called *rigid* if and only if the constructed logic yields only models $M = (\mathcal{W}, \{\mathcal{R}^i\}_{i\in I}, \{\mathcal{D}_w\}_{w\in\mathcal{W}}, \{\mathcal{I}_w\}_{w\in\mathcal{W}})$, such that for any two worlds $v, w \in \mathcal{W}$ it holds that $\mathcal{I}_v(c_\tau) = \mathcal{I}_w(c_\tau)$. ⌟

It is allowed to select different properties for any two constants, i.e. mixing rigid and flexible constants into the logic as desired.

2.2.4 Logical Consequence

The nature of HOML does not lead to only one meaningful notion of logical consequence as it is the case in classical logic. Fitting [23] describes a variant with locally and globally valid axioms: A conjecture is a logical consequence if and only if in all models in which the global axioms are valid the conjecture is true on all the worlds the local axioms are true.
In Gleißner et al. [28] two versions can be found: A conjecture is a local logical consequence if and only if all for all models and all worlds on which the axioms are true the conjecture is also true. Alternatively a conjecture is a global logical consequence if and only if for all models in which the axioms are valid the conjecture is also valid.
One can combine both approaches into two logical consequence relations in which the first one is actually Fitting's variant:

Definition 17 (Local logical consequence)
Let A^{global}, A^{local} be two sets of formuals and s_o be a formula. The term s_o is called a *local logical consequence* with global assumptions A^{global} and local assumptions A^{local} denoted [10] by $A^{global}, A^{local} \vDash^{HOML}_{local} s_o$ if and only if for all models M with $M \vDash^{HOML} A^{global}$ for all worlds w with $M, w \vDash^{HOML} A^{local}$ it holds that $M, w \vDash^{HOML} s_o$. ⌟

[10] Fitting uses the notation $A^{global} \vDash^{HOML}_{local} A^{local} \rightarrow s_o$.[22]

The second consequence relation differs from the first one only by quantifying over the worlds again when requiring s_o to be valid.

Definition 18 (Global logical consequence)
Let A^{global}, A^{local} be two sets of formulas and s_o be a formula. The term s_o is called a *global logical consequence* with global assumptions A^{global} and local assumptions A^{local} denoted by $A^{global}, A^{local} \models^{HOML}_{global} s_o$ if and only if for all models M with $M \models^{HOML} A^{global}$ and for all worlds w with $M, w \models^{HOML} A^{local}$ it holds that $M \models^{HOML} s_o$. ⌟

Summary

It is crucial to understand that with every single (or additional) choice of the semantic variations presented in this chapter, a unique logic is constructed. There are some exceptions when this is not true e.g. in the case of a symmetric and transitive accessibility relation since these properties imply its symmetry and euclideaness and vice versa. Apart from this the available options generate a vast quantity of different logics in which we choose

1. the number of $\Box^i$-operators,
2. the axiomatization of each $\Box^i$-operator using any combination of the nine axiom schemes,
3. the quantification semantics for each type $t \in \mathcal{T} \setminus \{o\}$ independently from the four options constant, varying, cumulative, decreasing,
4. rigid or flexible denotation for each constant,
5. one of the two logical consequence relations, and
6. local or global validity for every axiom of the problem.

3 Automation of Higher-Order Modal Logic

The automation of HOML can be achieved by using an approach called semantic embedding. During this process the semantics of a concrete variant of HOML are encoded in some other logic that offers enough expressivity to do so. In this case the target logic for the semantic embedding is HOL, specifically a variant of Church's simple type theory [19]. To that end, the meta-logical parts such as the accessibility relation, connectives and validity are formulated in HOL and all axioms and conjectures of the original HOML problem are converted by a translation procedure. The output of this approach can then be input of any HOL theorem proving system thus enabling the automation of HOML. This chapter merely summarizes the embedding of HOML in HOL. More detail and some completeness results can be found in the literature [26, 7, 12].

3.1 Classical Higher-Order Logic

Since the target logic of the encoding is HOL, a version of Church's *Simple Theory of Types* [19] is introduced that offers very similar features to HOML (and therefore typed λ-calculus) and is briefly introduced here. It is also a typed language and the set of HOL base types is given by $\{\boldsymbol{o}, \boldsymbol{\iota}, \boldsymbol{\mu}\}$ while the set of all HOL types $\boldsymbol{\mathcal{T}}$ is freely generated by creating function types from these base types, which will later be put to use as the (boolean) type for formulas, the individual domain and the set of worlds, respectively.
The terms of HOL are given by:

$$\boldsymbol{s}, \boldsymbol{t} \quad ::= \quad \boldsymbol{c_\tau} \mid \boldsymbol{X_\tau} \mid (\boldsymbol{\lambda X_\tau . s_\nu})_{\boldsymbol{\tau\nu}} \mid (\boldsymbol{s_{\tau\nu}\, t_\tau})_{\boldsymbol{\nu}}$$

where $\boldsymbol{X_\tau}$ is a *variable* from a countably infinite set $\boldsymbol{\mathcal{V}_\tau}$ of variable symbols for each type $\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}$ and $\boldsymbol{c_\tau} \in \boldsymbol{\Sigma_\tau}$ is a constant symbol of the signature $\boldsymbol{\Sigma} \equiv \bigcup_{\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}} \boldsymbol{\Sigma_\tau}$, that contains at least the primitive logical connectives for negation, disjunction, and, for each type, both equality and universal quantification. Boldface font is used to distinguish HOL terms and meta-logical definitions from those of HOML. HOL terms of type $\boldsymbol{o}$ are called *HOL formulas.*
A *frame* in HOL is a a collection $\{\boldsymbol{D_\tau}\}_{\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}}$ non-empty sets $\boldsymbol{\mathcal{D}_\tau}$, such that

1. $\boldsymbol{\mathcal{D}_o} \equiv \{\top, \bot\}$ is the domain of Booleans that represent truth $\top$ and falsehood $\bot$,
2. $\boldsymbol{\mathcal{D}_\iota}$,$\boldsymbol{\mathcal{D}_\mu}$ are two individual domains, and
3. $\boldsymbol{\mathcal{D}_{\tau\nu}}$ is the domain of total functions that map $\boldsymbol{\mathcal{D}_\tau}$ to $\boldsymbol{\mathcal{D}_\nu}$ for $\boldsymbol{\tau}, \boldsymbol{\nu} \in \boldsymbol{\mathcal{T}}$.

A *standard model* of HOL is a tuple $(\{\boldsymbol{D_\tau}\}_{\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}}, \boldsymbol{I})$ in which $\{\boldsymbol{D_\tau}\}_{\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}}$ comprises a HOL frame with $\boldsymbol{D_{\tau\nu}} \equiv \boldsymbol{D_\nu}^{\boldsymbol{D_\tau}}$ for $\boldsymbol{\tau}, \boldsymbol{\nu} \in \boldsymbol{\mathcal{T}}$ and $\boldsymbol{I}$ is an interpretation, mapping constant symbols $\boldsymbol{c_\tau}$ of any type $\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}$ to elements of their corresponding domain $\boldsymbol{\mathcal{D}_\tau}$. The constant symbols for disjunction, negation, universal quantification and equality have the usual denotation.
A *variable assignment* in HOL maps variables $\boldsymbol{X_\tau}$ of any type $\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}$ to elements of their corresponding domain $\boldsymbol{\mathcal{D}_\tau}$. The assignment $\boldsymbol{g}[\boldsymbol{d_\tau}/\boldsymbol{Y_\tau}]$ denotes the assignment that

is identical to $\boldsymbol{g}$, except for variable $\boldsymbol{Y_\tau}$ that is substituted by $\boldsymbol{d_\tau}$ with $\boldsymbol{d_\tau} \in \boldsymbol{\mathcal{D}_\tau}$ for a type $\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}$.
The *value* $\|\boldsymbol{u_\tau}\|^{\boldsymbol{M,g}}$ of a term $\boldsymbol{u_\tau}$ with respect to a model $(\{\boldsymbol{D_\tau}\}_{\tau \in \mathcal{T}}, \boldsymbol{I})$ under an assignment $\boldsymbol{g}$ is defined by

$$\begin{aligned}
\|\boldsymbol{X_\tau}\|^{\boldsymbol{M,g}} &\equiv \boldsymbol{g}(\boldsymbol{X_\tau}) \\
\|\boldsymbol{c_\tau}\|^{\boldsymbol{M,g}} &\equiv \boldsymbol{\mathcal{I}}(\boldsymbol{c_\tau}) \\
\|\,(\boldsymbol{\lambda X_\tau .\, s_\nu})_{\boldsymbol{\tau\nu}}\,\|^{\boldsymbol{M,g}} &\equiv f : \boldsymbol{D_\tau} \to \boldsymbol{D_\nu} \text{ with } y \overset{f}{\mapsto} \|\boldsymbol{s_\nu}\|^{\boldsymbol{M,g[X_\tau/y]}} \\
\|\,(\boldsymbol{s_{\tau\nu}\, t_\tau})_{\boldsymbol{\nu}}\,\|^{\boldsymbol{M,g}} &\equiv \|\boldsymbol{s_{\tau\nu}}\|^{\boldsymbol{M,g}}\big(\|\boldsymbol{t_\tau}\|^{\boldsymbol{M,g}}\big)
\end{aligned}$$

A *Henkin model* of HOL is a tuple $(\{\boldsymbol{D_\tau}\}_{\tau \in \mathcal{T}}, \boldsymbol{I})$ in which $\{\boldsymbol{D_\tau}\}_{\tau \in \mathcal{T}}$ comprises a HOL frame with $\boldsymbol{D_{\tau\nu}} \subseteq \boldsymbol{D_\nu}^{\boldsymbol{D_\tau}}$ for $\boldsymbol{\tau, \nu} \in \boldsymbol{\mathcal{T}}$, such that for any type $\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}$ the valuation function $\|\cdot\|^{\boldsymbol{M,g}}$ remains total (by the choice of the frame), and $\boldsymbol{I}$ is an interpretation, mapping constant symbols $\boldsymbol{c_\tau}$ of any type $\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}$ to elements of their corresponding domain $\boldsymbol{\mathcal{D}_\tau}$. Whenever HOL is of concern, Henkin models are considered for the remainder.
Notions of validity are:

$\boldsymbol{M}, \boldsymbol{g} \vDash^{HOL} \boldsymbol{s_o}$	A term $\boldsymbol{s_o}$ is *true* in model $\boldsymbol{M}$ under an assignment $\boldsymbol{g}$ if and only if $\|\boldsymbol{s_o}\|^{\boldsymbol{M,g}} \equiv \top$.
$\boldsymbol{M} \vDash^{HOL} \boldsymbol{s_o}$	A term $\boldsymbol{s_o}$ is *valid in model* $\boldsymbol{M}$ if and only if $\boldsymbol{M}, \boldsymbol{g} \vDash^{HOL} \boldsymbol{s_o}$ for all assignments $\boldsymbol{g}$.
$\vDash^{HOL} \boldsymbol{s_o}$	A term $\boldsymbol{s_o}$ is *valid* if and only if $\boldsymbol{M} \vDash^{HOL} \boldsymbol{s_o}$ for all models $\boldsymbol{M}$.

These definitions of $\vDash^{HOL}$ can also be used on sets of formulas in place of s_o in which the relations have to hold on all elements of this set.
The formula $\boldsymbol{s_o}$ is the *logical consequence* of a set $\boldsymbol{A}$ of HOL formulas, denoted $\boldsymbol{A} \vDash^{HOL} \boldsymbol{s_o}$, if and only if for all models $\boldsymbol{M}$ with $\boldsymbol{M} \vDash^{HOL} \boldsymbol{A}$ it holds that $\boldsymbol{M} \vDash^{HOL} \boldsymbol{s_o}$.

3.2 Semantic Embedding

The semantic embedding evolves around the type $\boldsymbol{\mu}$ that encodes the set of worlds $\mathcal{W}$ and is used to *lift* (cf. further below) expressions in order to depend on worlds. In particular HOML propositions will be associated with new HOL predicates of the world-dependent Boolean type $\boldsymbol{\mu o}$. This type is abbreviated as $\boldsymbol{\rho}$ for the remainder.

Accessibility relation. A new constant $\boldsymbol{r^i_{\mu\mu}}$ is introduced for each accessibility relation $\mathcal{R}^i \in \{\mathcal{R}^i\}_{i \in I}$ to the signature of HOL. The axiomatization of a $\Box^i$-operator can be realized by adding the desired axiom scheme from Table 1 about this operator to the problem. Alternatively it is possible utilize the Sahlqvist correspondences [44] of the axiomatizations in which each axiom corresponds to a mathematical property imposed on the accessibility relation as surveyed in Table 1 and adding them as axioms about the accessibility relation to the problem.

Logical connectives. Since the truth of a formula may be different on two distinct worlds the Boolean-typed connectives are furnished with a λ-abstraction on the worlds i.e. the truth of such a term is now world-dependent and therefore is typed with the afore mentioned type $\boldsymbol{\rho} \equiv \boldsymbol{\mu o}$. The encoding $\lceil\cdot\rceil$ for the set of HOML connectives is then given by

$$\begin{aligned}
\lceil\Box^i_{oo}\rceil &\equiv \;\; \boldsymbol{\Box^i_{\rho\rho}} &&:= \boldsymbol{\lambda S_\rho.\lambda W_\mu.}\forall \boldsymbol{V_\mu.}\ \neg(\boldsymbol{r^i}\ \boldsymbol{W}\ \boldsymbol{V}) \vee \boldsymbol{S}\ \boldsymbol{V}\\
\lceil\neg_{oo}\rceil &\equiv \;\; \neg_{\boldsymbol{\rho\rho}} &&:= \boldsymbol{\lambda S_\rho.\lambda W_\mu.}\ \neg(\boldsymbol{S}\ \boldsymbol{W})\\
\lceil\vee_{ooo}\rceil &\equiv \;\; \vee_{\boldsymbol{\rho\rho\rho}} &&:= \boldsymbol{\lambda S_\rho.\lambda T_\rho.\lambda W_\mu.}\ (\boldsymbol{S}\ \boldsymbol{W}) \vee (\boldsymbol{T}\ \boldsymbol{W})
\end{aligned}$$

The $\Box$-operator takes a world-dependent proposition $\boldsymbol{S}$, a world $\boldsymbol{W}$ from which this proposition should be evaluated and ensures it holds on all reachable worlds $\boldsymbol{V}$ exactly as the semantics of the operator commands. Negation and disjunction both just evaluate their world-dependent proposition(s) on the world they are passed.
All other connectives can either be defined within HOML as exemplified in section 2.1 and therefore are encoded implicitly or the encoding can be made explicit:

$$\begin{aligned}
\lceil\Diamond^i_{oo}\rceil &\equiv \;\; \boldsymbol{\Diamond^i_{\rho\rho}} &&:= \boldsymbol{\lambda S_\rho.\lambda W_\mu.}\exists \boldsymbol{V_\mu.}\ (\boldsymbol{r^i}\ \boldsymbol{W}\ \boldsymbol{V}) \wedge \boldsymbol{S}\ \boldsymbol{V}\\
\lceil\wedge^i_{ooo}\rceil &\equiv \;\; \wedge^{\boldsymbol{i}}_{\boldsymbol{\rho\rho\rho}} &&:= \boldsymbol{\lambda S_\rho.\lambda T_\rho.\lambda W_\mu.}\ (\boldsymbol{S}\ \boldsymbol{W}) \wedge (\boldsymbol{T}\ \boldsymbol{W})\\
\lceil\supset^i_{ooo}\rceil &\equiv \;\; \supset^{\boldsymbol{i}}_{\boldsymbol{\rho\rho\rho}} &&:= \boldsymbol{\lambda S_\rho.\lambda T_\rho.\lambda W_\mu.}\ \neg(\boldsymbol{S}\ \boldsymbol{W}) \vee (\boldsymbol{T}\ \boldsymbol{W})
\end{aligned}$$

Quantifiers. The encoding $\lceil\Pi^\tau\rceil$ of universal quantification of type $\tau \in \mathcal{T}$ depends on whether this quantification is intended to have constant domain or varying domain semantics. Let $\boldsymbol{\Pi}^{\boldsymbol{\tau},\mathrm{c}}$ and $\boldsymbol{\Pi}^{\boldsymbol{\tau},\mathrm{va}}$ denote the encoding of a constant and varying domain quantification term, respectively, defined by

$$\begin{aligned}
\boldsymbol{\Pi}^{\boldsymbol{\tau},\mathrm{va}}_{\boldsymbol{(\tau\rho)\rho}} &:= \boldsymbol{\lambda P_{\tau\rho}.\lambda W_\mu.}\forall \boldsymbol{X_\tau.}\ \neg(\mathbf{eiw}^{\boldsymbol{\tau}}_{\boldsymbol{\tau\mu o}}\ \boldsymbol{X}\ \boldsymbol{W}) \vee (\boldsymbol{P}\ \boldsymbol{X}\ \boldsymbol{W})\\
\boldsymbol{\Pi}^{\boldsymbol{\tau},\mathrm{c}}_{\boldsymbol{(\tau\rho)\rho}} &:= \boldsymbol{\lambda P_{\tau\rho}.\lambda W_\mu.}\forall \boldsymbol{X_\tau.}\ \boldsymbol{P}\ \boldsymbol{X}\ \boldsymbol{W}
\end{aligned}$$

The idea behind simulating the world-dependent domain collections $\mathcal{D}_w \in \{\mathcal{D}_w\}_{w\in\mathcal{W}}$ of HOML is creating supersets of all possible elements of a certain type and guarding the access to this set by predicates that specify whether an element indeed exists on some world. Such set for a type $\tau \in \mathcal{T}$ is defined by $\{d \in D^w_\tau | D^w_\tau \in \{\mathcal{D}_w\}_{w\in\mathcal{W}}, w \in \mathcal{W}\}$ and consists of all elements of this type which are contained in any domain D^w_τ of any domain collection $\{\mathcal{D}_w\}_{w\in\mathcal{W}}$. These sets are postulated to be equal to the domain collections $\boldsymbol{D_\tau} \in \boldsymbol{\{D_\tau\}_{\tau\in\mathcal{T}}}$ in HOL of their corresponding type $\boldsymbol{\tau}$. In the case of varying domain quantification the guarding predicate $\mathbf{eiw}^{\boldsymbol{\tau}}_{\boldsymbol{\tau\mu o}}$ prevents the evaluation of $\boldsymbol{P}$ for any substitution $\boldsymbol{g}[\lceil s_\tau\rceil/\boldsymbol{X_\tau}]$ of an assignment g if $s_\tau \notin D^W_\tau$ and permits it otherwise as required. Such a guarding predicate is added to the signature of HOL for every type $\tau \in \mathcal{T}$. Constants in HOML must denote on any world hence their existence on all worlds has to be made explicit by adding an axiom for each constant $\boldsymbol{c_\tau}$:

$$\forall \boldsymbol{W_\mu.}\ \mathbf{eiw}^{\boldsymbol{\tau}}_{\boldsymbol{\tau\mu o}}\ \boldsymbol{c_\tau}\ \boldsymbol{W}$$

For constant domain semantics the contents of all world-dependent domains D^w_τ are the same hence the guarding predicate is simply omitted.
In the cases of cumulative or decreasing quantification semantics, varying quantifiers are employed and an appropriate axiom is added to the problem to ensure the expected domain structure across the worlds. The axioms are, respectively,

$$\forall \boldsymbol{W_\mu}. \forall \boldsymbol{V_\mu}. \forall \boldsymbol{X_\tau}.\ (\mathbf{eiw}^{\boldsymbol{\tau}}_{\boldsymbol{\tau\mu o}}\ \boldsymbol{X}\ \boldsymbol{V} \wedge \boldsymbol{r^i}\ \boldsymbol{V}\ \boldsymbol{W}) \supset \mathbf{eiw}^{\boldsymbol{\tau}}_{\boldsymbol{\tau\mu o}}\ \boldsymbol{X}\ \boldsymbol{W}$$
$$\forall \boldsymbol{W_\mu}. \forall \boldsymbol{V_\mu}. \forall \boldsymbol{X_\tau}.\ (\mathbf{eiw}^{\boldsymbol{\tau}}_{\boldsymbol{\tau\mu o}}\ \boldsymbol{X}\ \boldsymbol{W} \wedge \boldsymbol{r^i}\ \boldsymbol{V}\ \boldsymbol{W}) \supset \mathbf{eiw}^{\boldsymbol{\tau}}_{\boldsymbol{\tau\mu o}}\ \boldsymbol{X}\ \boldsymbol{V}$$

The axiom for cumulative quantification semantics states that if $\boldsymbol{X}$ exists in some world $\boldsymbol{V}$, it also exists in all reachable worlds. Decreasing quantification semantics works along the other direction of the accessibility relation.
Existential quantification, like the additional connectives, can also be encoded in a direct manner:

$$\exists^{\boldsymbol{\tau},\mathrm{va}}_{\boldsymbol{(\tau\rho)\rho}} := \boldsymbol{\lambda P_{\tau\rho}}. \boldsymbol{\lambda W_\mu}. \exists \boldsymbol{X_\tau}.\ (\mathbf{eiw}^{\boldsymbol{\tau}}_{\boldsymbol{\tau\mu o}}\ \boldsymbol{X}\ \boldsymbol{W}) \wedge (\boldsymbol{P}\ \boldsymbol{X}\ \boldsymbol{W})$$
$$\exists^{\boldsymbol{\tau},\mathrm{c}}_{\boldsymbol{(\tau\rho)\rho}} := \boldsymbol{\lambda P_{\tau\rho}}. \boldsymbol{\lambda W_\mu}. \exists \boldsymbol{X_\tau}.\ \boldsymbol{P}\ \boldsymbol{X}\ \boldsymbol{W}$$

The constant and varying domain cases for existential quantification are analogous to universal quantification.

An alternative embedding for quantification in constant, cumulative and decreasing settings can be realized through the use of the so-called Barcan (BF) and Converse Barcan Formula (CBF). In the first-order domain, adding BF as an axiom scheme when simultaneously employing varying quantification semantics orders the individual domain as decreasing quantification semantics require while CBF imposes cumulative domains [23]. A similar result might probably hold for non-first-order domains, too, but has not been proven yet. Adding both BF and CBF renders the domains decreasing and cumulative, hence constant. The schemes BF and CBF for type $\tau \in \mathcal{T}$ are given by

$$\Box\, \forall X_\tau.\ P_{\tau o}\ X \supset \forall X_\tau.\ \Box P_{\tau o}\ X$$
$$\forall X_\tau.\ \Box P_{\tau o}\ X \supset \Box\, \forall X_\tau.\ P_{\tau o}\ X$$

respectively.

There also exists one further encoding for constant, cumulative and decreasing quantification semantics in modal system S5 which is called S5U in the remainder. In that case the accessibility relation can be seen as an universal relation between possible worlds [40]. That means all worlds are connected to each other, thereby eliminating the need of checking if two worlds are connected. When exploiting this property, encoding the accessibility relation becomes no longer necessary, hence all occurrences of the accessibility relation are stripped from the encoding: First, modal operators are simplified, not checking for connected worlds, and can be defined as

$$\lceil \Box^i_{oo} \rceil \equiv \boldsymbol{\Box^i_{\rho\rho}} := \boldsymbol{\lambda S_\rho}. \boldsymbol{\lambda W_\mu}. \forall \boldsymbol{V_\mu}.\ \boldsymbol{S}\ \boldsymbol{V}$$

Second, all quantifiers can be embedded as constant quantifiers because once an object of a constant, cumulative or decreasing domain exists in some world, it has to exist in any other world since all worlds are connected and the domains cannot decrease or increase, depending on the quantification semantics.

Embedding the problem. For each type $\tau \in \mathcal{T}$ the encoding (or type-lifting) $\lceil \tau \rceil$ is defined as

$$\begin{aligned}
\lceil o \rceil &\equiv \boldsymbol{\rho} := \boldsymbol{\mu o} \\
\lceil \iota \rceil &\equiv \boldsymbol{\iota} \\
\lceil \tau\nu \rceil &\equiv \lceil \tau \rceil \lceil \nu \rceil
\end{aligned}$$

for $\tau, \nu \in \mathcal{T}$.
The definition of $\lceil \cdot \rceil$ can now be extended to all HOML terms thereby mimicking (with the contribution of the previous definitions and auxiliary axioms) the behavior of the valuation function and the general model structure by

$$\begin{aligned}
\lceil c_\iota \rceil &\equiv \boldsymbol{c}_{\lceil \mu\iota \rceil} && \text{if } c_\iota \text{ is a flexible constant} \\
\lceil c_\tau \rceil &\equiv \boldsymbol{c}_{\lceil \tau \rceil} && \text{if } c_\tau \text{ is a rigid constant} \\
\lceil X_\tau \rceil &\equiv \boldsymbol{X}_{\lceil \tau \rceil} \\
\lceil \lambda X_\tau .\, s_\nu \rceil &\equiv \boldsymbol{\lambda} \lceil X_\tau \rceil \boldsymbol{.}\, \lceil s_\nu \rceil \\
\lceil s_{\tau\nu}\, t_\tau \rceil &\equiv \lceil s_{\tau\nu} \rceil\, \lceil t_\tau \rceil
\end{aligned}$$

for $\tau, \nu \in \mathcal{T}$. Now all products of the grammar of HOML are mapped to a (bold) HOL counterpart.
For the notions of validity and logical consequence the meta-logical functions $\lfloor \cdot \rfloor$ and $\boldsymbol{\mathcal{A}}(\cdot)$ are introduced:

$$\begin{aligned}
\lfloor \boldsymbol{s_\rho} \rfloor &:= \forall \boldsymbol{W_\mu} .\, \boldsymbol{s}\, \boldsymbol{W} \\
\boldsymbol{\mathcal{A}}(\boldsymbol{s_\rho}) &:= \boldsymbol{s}\, \boldsymbol{w}^{\textbf{actual}}
\end{aligned}$$

where $\boldsymbol{w}^{\textbf{actual}}$ is an uninterpreted constant of the HOL signature. Both functions take a world-dependent proposition and map it to a value of type $\boldsymbol{o}$ i.e. to truth or falsehood in HOL.
All global axioms have to hold on every world hence they are (grounded) by the function $\lfloor \cdot \rfloor$ which ensures the global axioms are globally valid by quantifying over all worlds as required by both notions of logical consequence. Local axioms are grounded by applying $\boldsymbol{\mathcal{A}}(\cdot)$. After filtering for all models in which the global axioms hold true, $\boldsymbol{\mathcal{A}}(\cdot)$ implicitly filters for all worlds of those models on which all local axioms hold true. This set of filtered worlds is represented by the uninterpreted world constant $\boldsymbol{w}^{\textbf{actual}}$.

The definition of $\lfloor \cdot \rfloor$ allows for an easy encoding of axiom schemes as required for axiom schemes on the modal operators if not using accessibility relation properties, and implementing BF and CBF as axiom schemes for the alternative embedding of constant, cumulative and decreasing quantification semantics. Exemplary, the axiom scheme BF for type $\tau \in \mathcal{T}$ that reads

$$\Box\, \forall X_\tau.\, P_{\tau o}\, X \supset \forall X_\tau.\, \Box P_{\tau o}\, X$$

can be encoded as

$$\forall \boldsymbol{P_{\tau(\mu o)}}.\, \lfloor \Box\, \forall X_\tau.\, P_{\tau o}\, X \supset \forall X_\tau.\, \Box P_{\tau o}\, X \rfloor$$

where $P_{\tau o}$ will be recursively translated to $\boldsymbol{P_{\tau(\mu o)}}$ by the grounding function $\lfloor \cdot \rfloor$.
If a conjecture is supposed to be a local logical consequence, it is grounded by $\boldsymbol{\mathcal{A}}(\cdot)$ since it has to hold true on all filtered worlds represented by $\boldsymbol{w}^{\textbf{actual}}$. In the case of global logical consequence, the conjecture is grounded by $\lfloor \cdot \rfloor$ as the consequence should be globally valid in all filtered models. More formally, we have that

$$A^{global}, A^{local} \vDash^{HOML}_{global} s$$

if and only if

$$\{\lfloor \lceil a \rceil \rfloor | a \in A^{global}\} \cup \{\boldsymbol{\mathcal{A}}(\lceil a \rceil) | a \in A^{local}\} \vDash^{HOL} \lfloor \lceil s \rceil \rfloor$$

for global logical consequence, and

$$A^{global}, A^{local} \vDash^{HOML}_{local} s$$

if and only if

$$\{\lfloor \lceil a \rceil \rfloor | a \in A^{global}\} \cup \{\boldsymbol{\mathcal{A}}(\lceil a \rceil) | a \in A^{local}\} \vDash^{HOL} \boldsymbol{A}(\lceil s \rceil)$$

for local logical consequence. A proof for global consequence for the first-order variant of modal logic with global axioms and constant domain semantics was presented in *Benzmüller and Paulson* [7]. Proofs for other variants do not exist yet.

4 The Modal Embedding Tool

The Modal Embedding Tool (MET) [11] is an implementation of the shallow semantic embedding approach for HOML as presented in sect. 3.

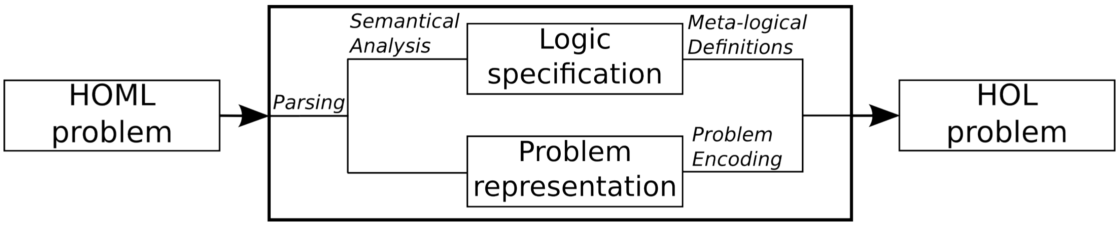

Figure 1: Bird's eye perspective of the automated embedding process

Figure 1 illustrates the workflow of the MET. Firstly, it analyzes the logic specification i.e. semantics description according to sect. 4.1 and identifies meta-logical definitions e.g. embedded quantifiers, accessibility relations, etc. and axioms e.g. cumulative domain restrictions, accessibility relation properties, etc., that are required. Subsequently, the original problem representation in HOML is modified by exchanging connectives, quantifiers and types by their encoded counterpart and by grounding all formulas. Finally both modified original representation and meta-logical definitions and axioms are combined resulting in a corresponding HOL problem, ready to be shipped to any higher-order prover understanding HOL. More detail on the implementation can be found in Gleißner [26].
The remainder of this section is intended to describe the input language and capabilities of the MET.

4.1 Input Language

The input language of the MET is a conservative extension of the TPTP dialect THF [8] while the output language again is THF. It is the de-facto standard for higher-order logic representation and reasoning and is supported by most HOL reasoners, including Satallax [17], Leo III [48], Nitpick [15] and many others. The purpose of this input language is to foster a standardized format for non-classical logics within the TPTP. Previous work [57] already dealt with such an extensions and was subsequently improved to yield a TPTP standard proposal that is still under development. [12]

Syntax. THF supports all syntactic features of HOL including lambda abstraction and function application. It also supports and defines functions that bear meaning and therefore allow for an easy adaptation for the additional elements of the HOML syntax, i.e. $\Box$- and $\Diamond$-operators. Mono-modal problems use the defined functions `$box` and

[11] Available at `https://github.com/leoprover/embed_modal`.
[12] The current status of this proposal can be found on `http://www.tptp.org/TPTP/Proposals/LogicSpecification.html`.

`$dia` for necessity and possibility hence a HOML formula $\exists X_\iota .\ \Box(p\ X)$ corresponds to `? [X:$i]: ($box @ (p @ X))`.
In multi-modal logics the defined functions `$box_int` and `$dia_int` can be applied to an integer that serves as an index for the $\Box^i$- and $\Diamond^i$-operators, respectively. Exemplary, $\exists X_\iota .\ \Box^1(p\ X)$ corresponds to `? [X:$i]: ($box_int @ 1 @ (p @ X))`.
An alternative input syntax for multiple modalities can be achieved through rank-1 polymorphic types offered by the THF extension TH1 [34]. The defined function `$box_P` (analogously `$dia_P`) of type `!>[T:$tType]:(T > $o > $o)` is added to the signature. It takes as arguments a type `T` employing the type binder `!>` that enables rank-1 polymorphism, an object of type `T`, a boolean formula, and emits a boolean formula. This permits creating different types of boxes and indexing them with objects of that type as follows:

```
thf( human_box_type, type, human: $tType ).
thf( alice_type, type, alice: human ).
thf( myproposition_type, type, myproposition: $o ).
thf( 1, axiom, $box_P @ human @ alice @ myproposition ).
```

Modality types or instances of polymorphic modalities can also be abbreviated:

```
thf( human_box_modality_type, type, human_box : (human > $o > $o) ).
thf( human_box_modality_def, definition, human_box = ($box_P @ human) ).
thf( 1, axiom, human_box @ alice @ myproposition ).
thf( alice_box_modality_type, type, alice_box : ($o > $o) ).
thf( alice_box_modality_def, definition, alice_box = ($box_P @ human @ alice) ).
thf( 2, axiom, alice_box @ myproposition ).
```

This polymorphic approach groups modalities (by type, here `human`) which is utilized in the next paragraph for simultaneously specifying the same semantics for every modality of one type. Moreover, it offers a more natural indexing mechanism by allowing an actual string for this purpose.

Semantics specification. Specifying the semantics for the modal logic that was intended for a HOML problem takes place in a `logic` statement i.e. a statement, that yields the role `logic`. The keyword `$modal` indicates the beginning of a list of semantics options for Kripke-like logics. This list has to include the keywords `$constants`, `$quantification`, `$consequence` and `$modalities`. Every such keyword is followed by an assignment operator `:=` and either a default value or a comma-separated list of entries. A simple logic specification that uses one default value for each semantics dimension (of HOML) may look as follows:

```
thf( my_s5, logic, ( $modal := [
     $constants      := $rigid,
     $quantification := $constant,
     $consequence    := $local,
     $modalities     := $modal_system_S5
] ) ).
```

In this example constants are assumed rigid, quantification semantics of all domains are considered constant, all axioms are locally valid and the logical consequence is local, too, and, all occurring modalities possess the properties defined by modal system S5.
This format also allows for more involved descriptions i.e. it grants control on the designation of every single constant, the quantification semantics for each type, local or global validity of assumptions and hypothesis, and, the axiomatization of every modal operator, separately. At the same time it offers default values for all parts of the semantics that have not been explicitly addressed. The full extent of logic specification features is show-cased in the following example:

```
thf( mydomain_type , type , ( mynewdomain : $tType ) ).
thf( myconstant_declaration , type , ( myconstant : $i ) ).
thf( myaxiom , axiom , ( ! [X:$o]: ( ($box_int @ 3 @ X) => ($box_int @ 1 @ X) ) ) ).
thf( myhypothesis , conjecture , ( ! [X:$o]: ($box_int @ 1 @ ($box_int @ 2 @ X) ) ) ).
thf( animal_box_type, type, animal: $tType ).
thf( human_box_type, type, human: $tType ).
thf( alice_type, type, alice: human ).
thf( fancy_modal_logic , logic , ( $modal := [
     $constants      := [ $rigid,
                          myconstant := $flexible ],
     $quantification := [ $constant,
                          mynewdomain := $varying ],
     $consequence    := [ $local,
                          myaxiom := $global,
                          myhypothesis := $global ],
     $modalities     := [ $modal_system_S5,
                          $box_int @ 1 := $modal_system_T,
                          $box_int @ 2 := [$modal_axiom_4, $modal_axiom_C],
                          $box_P @ human @ alice := $modal_system_S4,
                          $box_P @ animal := $modal_system_D ]
] ) ).
```

Here, all constants have a rigid designation except for constant `myconstant` which is flexible, the quantification semantics comprise constant domains per default but domain `mydomain` is considered varying, and, all assumptions are local but axiom `myaxiom` which is globally valid, also the conjecture `myhypothesis` is one of a global consequence. Modal operator $\Box^1$ possesses the properties of modal system T and axiom schemes 4 and C are attributed to $\Box^2$. Modality `$box_P @ human @ alice` possesses the semantics of modal system S4 and all modalities of type `animal` have modal system D properties. All other modalities have S5 properties assigned per default.
This exemplary logic specification demonstrates for each semantics dimension the assignment of a default value (first item of the list) for all concerned elements of this dimension except for those explicitly listed after the default value. These elements are assigned a value on their own using the assignment operator `:=` after the objects name, followed by the desired value. It is not allowed to specify the semantics for both a type of modalities and one modality of this type e.g. `$box_P @ human` and `$box_P @ human @ alice` are not allowed to occur in the specification simultaneously. For practical reasons it is not permitted to use any definitions e.g. an abbreviation for `$box_P @ human @ alice` or

`$box_P @ human` (cf. syntax paragraph before) within the specification, since otherwise the effort for gathering all information about the semantics increases.
Values supported by the logic specification format are

- `$constants`: {`$rigid`, `$flexible`}
- `$quantification`: {`$constant`, `$cumulative`, `$decreasing`, `$varying`}
- `$consequence`: {`$local`, `$global`}
- `$modalities`: `$modal_system_X` for
  `X` $\in$ {K, KB, K4, K5, K45, KB5, D, DB, D4, D5, D45, T, B, S4, S5}
  or a list of `$modal_axiom_Y` with `Y` $\in$ {T, B, D, 4, 5, CD, $\Box$M, C4, C}

4.2 Usage and Scope

The MET is a tool for the command line interface (CLI). It can be invoked by

```
java -jar embed.jar -i infile -o outfile
```

where `infile` and `outfile` specify the input file in TPTP THF syntax with modal extension as presented in sect. 4.1 and the output file which will have TPTP THF format, respectively. `$box` and `$box_int` but not `$box_P` are supported by the MET.
Available CLI parameters and other options presented in this section are summarized in appendix D.

Supported semantics. The MET implements the full range of semantics alternatives for HOML with only few exceptions. It supports mono-modal reasoning for constant, cumulative, decreasing and varying domain semantics individually for each type. Assumptions and hypothesis can be declared locally or globally valid where assorted choices are possible, and, modalities might take on any of the 15 modal systems or nine axioms listed in sect. 4.1. Only flexible constants have not been implemented hence all constants are considered rigid. Multi-modal reasoning includes the same options employing integer-indexed box operators with the exception of cumulative and decreasing domains.

Alternative encodings. As displayed in sect. 3.2, there are multiple ways to encode semantics. The MET allows the user to pass a comma-separated list of transformation parameters using the CLI option `-t` for this purpose.
There exist three different ways for embedding modalities: Firstly, the used properties can be selected by explicitly choosing the axiomatization of the modal operators since some systems may can expressed by more than one set of rules e.g. modal system S5 can be described by axiom schemes {K,T,5},{K,T,B,4} and many others. Such explicit axiomatization is enabled through a system specific extension on the keywords of the logic specification e.g. `$modal_system_S5` can be substituted by `$modal_system_S5_KT5` for axiomatization {K,T,5}, etc. Moreover, the encoding S5U for modal system S5 can be enabled using the keyword `$modal_system_S5U`. Secondly, the modal operators' attributes

can be encoded with axiom schemes, or, as third option, postulating properties on the accessibility relations, by passing the parameters `syntactic_modality_axiomatization` or `semantic_modality_axiomatization`, respectively.
Alternatives for quantification semantics offered by the MET for cumulative / decreasing domain restrictions are either involving properties on the domain structure attaching parameters `semantic_cumulative_quantification` / `semantic_decreasing_quantification`, or, employing CBF/BF with parameters `syntactic_cumulative_quantification` / `syntactic_decreasing_quantification`. Also, constant domains may be encoded using constant quantifiers (parameter `semantic_constant_quantification`) or through the use of varying quantifiers with CBF+BF (parameter `syntactic_constant_quantification`) instead.

5 Evaluation

In this section, the MET is evaluated with respect to correctness, completeness and performance using the QMLTP library [42] that consists of 580 first-order mono-modal problems. To that end, three higher-order provers, paired with different encodings provided by the MET, and one of the most sophisticated native modal logic provers are examined for their performance. First, the experimental set-up is outlined, subsequently the detailed evaluation results are presented and discussed.

5.1 Experimental Set-up

The experimental set-up divides itself into a paragraph about the dataset including its preparation and a description of the hardware and software environment in which the benchmarks take place.

Dataset. All Problems of the QMLTP are written in a variant of the TPTP dialect FOF and have already been translated to TPTP THF with modal extension by Gleißner [26]. Unfortunately some of these problems had missing parentheses and missing axiom issues, that could partially be fixed (cf. Table 3). Missing parentheses issues could be identified applying an ANTLR-generated parser [26]. The dataset was filtered for problems that possess equalities but do not contain an appropriate axiomatization as required by the QMLTP. Furthermore all occurrences of the symbol "=" have been substituted by the symbol "qmltpeq" since in the QMLTP "=" is an uninterpreted symbol while it represents native equality in TPTP. The resulting 566 QMLTP problems that have been transformed in this manner form the basis of this evaluation. [13]

Hardware and software environment. The evaluation was performed using the StarExec [49] compute cluster. Each node of this cluster runs an Intel(R) Xeon(R) CPU E5-2609 2.40GHz CPU with 256 GB of RAM. The wall clock time (WC) was limited to 480 seconds and the computing time (CPU) was limited to 240 seconds. This configuration is similar to other competitions e.g. the CADE ATP System Competition (CASC) [50].
First, the most advanced [5] native first-order modal logic prover MleanCoP [41] was benchmarked against the dataset. This prover supports local logical consequence, rigid constants, modal systems D, T, S4 and S5, and quantification semantics constant, cumulative and varying in the first-order domain. Hence for comparability, the benchmark involves all possible combinations of these semantics choices. Second, the state-of-the-art higher-order theorem provers Leo-III [48] and Satallax [16] as well as the model finder Nitpick [15] are combined with the MET for the assessment of the MET's performance.

[13]The dataset is available at `https://github.com/TobiasGleissner/QMLTP`.

Problem	**Issue**
APM004+1.p	parentheses (fixed)
APM007+1.p	symbol ”qmltpeq” without axiomatization
SYM035+1.p	parentheses (fixed)
SYM054+1.p	symbol ”qmltpeq” without axiomatization
SYM055+1.p	symbol ”=” without axiomatization
SYM056+1.p	symbol ”=” without axiomatization
SYM057+1.p	symbol ”=” without axiomatization
SYM058+1.p	symbol ”qmltpeq” without axiomatization
SYM064+1.p	symbol ”=” without axiomatization
SYM068+1.p	symbol ”=” and ”qmltpeq” without axiomatization
SYM069+1.p	symbol ”qmltpeq” without axiomatization
SYM070+1.p	symbol ”qmltpeq” without axiomatization
SYM071+1.p	symbol ”qmltpeq” without axiomatization
SYM072+1.p	symbol ”qmltpeq” without axiomatization
SYM082+1.p	parentheses (fixed)
SYM085+1.p	symbol ”=” without axiomatization
SYM113+1.p	parentheses (fixed)
SYM114+1.p	parentheses (fixed)
SYM122+1.p	parentheses (fixed)
SYM163+1.p	parentheses (fixed)

Table 3: Issues with problems in the QMLTP library

The following synopsis lists details on all employed reasoning systems:

- MleanCoP version 1.3
- MET + Leo-III in a prerelease of version 1.3 in cooperation with the external reasoners E 2.3 and CVC4 1.7, It runs on a Java Runtime Environment 8 with options `-Xss128m -Xmx4g -Xms1g`
- MET + Satallax version 3.3 built with OCaml 4.05.0
- MET + Nitpick of Isabelle release candidate 2018

All different modal logic encodings offered by the MET for MleanCoP's supported semantics have been tested. This includes the encoding of different modal systems by either imposing restrictions on the accessibility relation, adding axiom schemes about the modal operator or using the optimized encoding S5U. Quantification encodings include constant quantifiers (without an eiw-predicate) for constant domains as well as varying quantifiers for varying, cumulative, constant domains with no modification, CBF or domain restriction axiom, CBF+BF or domain restriction axioms, respectively. A comprehensive summary of all modal system and quantification encodings and their abbreviations that are used during the evaluation is listed in Tables 4a and 4b. For details on these encodings see sect. 3.2. Note that the application of the MET always counts towards reasoning time (WC and CPU), since reencoding a problem is regarded as a part of the reasoning process to solve the problem.

X_{sem}	modalities are specified by imposing restrictions on the accessibility relation
X_{syn}	modalities are specified by adding axiom schemes about the modalities
S5U	the optimized encoding for modal system S5 is used

(a) Modal system encodings, X ∈ {D,T,S4,S5}

vary	uses varying quantifiers
cumul_{syn}	uses varying quantifiers and CBF
cumul_{sem}	uses varying quantifiers and cumulative domain restriction axiom
const_{syn}	used varying quantifiers and CBF and BF
const_{sem}	uses constant quantifiers

(b) Quantification encodings

Table 4: Encodings offered by the MET

5.2 Results

The analysis of the benchmark assesses the merits of different encodings and the performance in terms of solved problems and reasoning time of all involved reasoning systems. In particular the findings are subdivided into

- development cost,
- incompleteness issue,
- comparison of supported semantics by the MET and MleanCoP
- comparison of different encodings,
- comparison of solved theorems by the MET and MleanCoP,
- comparison of problems found counter-satisfiable by the MET and MleanCoP, and
- comparison of CPU and WC usage by the MET and MleanCoP,

which are treated in separate paragraphs in that order. The term HOany refers to the combined result of all higher-order provers using the MET with the best encoding, i.e. most problems solved, for the semantics in question. For proving theorems, HOany includes only results of Leo-III and Satallax, since Nitpick is a model finder. As for comparing capabilites in counter-model finding Leo-III and Satallax are usually not explicitly listed since they perform poorly at this task and Nitpick always finds a counter-model for any problem that was found counter-satisfiable by Leo-III or Satallax. Since the MET is a preprocessing tool and therefore can be easily combined with any higher-order reasoning system, it is justified to compare HOany and the native prover MleanCoP. Even more so, today's computer hardware comes with several cores which permits trying different provers simultaneously on one machine without sacrificing wall clock time.

Miscellaneous. For any encoding of the dataset, the MET did not produce any errors. Furthermore, there have not been any unexpected output e.g. a surprising SZS status or other errors by any reasoning system employing the MET or the prover MleanCoP. Finally, no soundness problems have occurred i.e. no higher-order prover paired with the MET reports the status theorem for a problem (in any encoding) if the QMLTP library or the reference system MleanCoP finds the problem to be counter-satisfiable.

Development cost. The development time required for implementing a semantic embedding (for all the variants of HOML) is substantially less than devising a specialized reasoner. Its development involves only a fraction of the steps necessary for building an ATP, essentially summing up to parsing and rewriting. It is a matter of folklore that it takes roughly two years to build a reasoning system that is as as good as Otter [38] and Otter has not been developed since 2003. Moreover, examining the participants and results of CASC [50, 52], the most successful provers have been developed and refined over many years. This indicates that it generally takes several years to implement a high-performing ATP system, whereas a semantic embedding procedure can be implemented much faster. In conclusion the semantic embedding approach provides a shortcut to reduce the software development cost and deficit in non-classical reasoning tools.

Incompleteness issue. Experiments involving the (counter-)model finder Nitpick combined with the MET show that there is an incompleteness issue with encodings that make use of varying quantifiers. That is, for problems that are actually classified as theorems, a higher-order prover paired with the MET may find such problems to be counter-satisfiable. This affects the encodings $\text{const}_{\text{syn}}$, $\text{cumul}_{\text{syn}}$, vary and supposedly (but without experimental evidence) decr_{syn}. An analysis of counter-models found by Nitpick suggests an inproper guarding (cf. eiw predicate sect. 3.2, paragraph on quantifiers) against objects that should not exist in the corresponding domain (of HOML) hence not in any (counter-)model. This effect could be observed at least when using functions that take elements of the individual domain as arguments and emit an individual as Figure 2 indicates. It shows a counter-model snippet provided by Nitpick for the problem GLC454+1 encoded with modal system S5 and quantification semantics $\text{cumul}_{\text{sem}}$. Line 2 states the existence of two objects, i_1 and i_2 of type ι (two individuals), and one object of type mworld (one world). Lines 13 to 16 refer to the eiw-predicate that should prevent the use of non-existing individuals. Line 15 and 16 assert $eiw\ i_1 \equiv \top$ and $eiw\ i_2 \equiv \bot$ i.e. the first individual i_1 does exist and the second individual i_2 does not exist with respect to the guarding predicate. However the user-defined 2-ary function *and* (lines 6 to 11) may take i_2 as an argument (lines 13 and 15) and can even emit the non-existent object i_2 (line 15). It is unclear whether guarding functions against non-existent objects will suffice to overcome the whole incompleteness. Probably a (computer-verified) attempt to prove the completeness of a mended embedding may shed light on this matter.

```
Nitpick found a counterexample for
card TPTP_Interpret.ind = 2 and card bnd_mworld = 1:
Skolem constant:
??.bnd_mbox.V = b\<^sub>1
Constants:
bnd_and =
(\<lambda>x. _)
(i\<^sub>1 := (\<lambda>x. _)
(i\<^sub>1 := i\<^sub>1, i\<^sub>2 := i\<^sub>1),
i\<^sub>2 := (\<lambda>x. _)
(i\<^sub>1 := i\<^sub>1, i\<^sub>2 := i\<^sub>2))
[...]
bnd_eiw__o__d_i_c_ =
(\<lambda>x. _)
(i\<^sub>1 := (\<lambda>x. _)(b\<^sub>1 := True),
i\<^sub>2 := (\<lambda>x. _)(b\<^sub>1 := False))
[...]
```

Figure 2: Counter-model snippet by Nitpick demonstrating the occurrence of non-existing HOML objects when using varying quantifiers in the encoding

Comparison of supported modal logics. The native prover MleanCoP offers first-order quantification, multiple modalities, a complete calculus, four modal systems, three types of quantification semantics and one logical consequence, amounting to 11 different settings for mono-modal semantics. The MET offers higher-order quantification, multiple modalities, an incomplete transformation procedure, 15 modal systems, four types of quantification semantics, two options for logical consequence and mixing of local and global assumptions and hypothesis, amounting to 116 different settings for mono-modal semantics.
Clearly the MET is tremendously more versatile than MleanCoP. It supports 10 times more semantics settings, has higher-order quantification and allows for a mixture of local and global assumptions. The only drawback occurs when the assignment consists of finding counter-models since the MET supports only constant domain semantics across all modal systems as well as constant, cumulative and decreasing domain semantics for modal system S5 for performing this task.

Comparison of different encodings. Tables 5, 6 and 7 account for the individual performance of each reasoning system that employs the MET i.e. its solved problems and reasoning time, the performance against other embedding provers (only Leo-III's and Satallax' measurements) and against the native prover MleanCoP. Each row is associated with the encoding of a modal system and the encoding of quantification semantics as listed in Tables 4a and 4b. The results of each reasoning system with respect to the encoding are counted according to their status including theorem (THM), counter-satisfiability (CSA), gave-up (GUP), timeout (TMO) and total solutions (Σ) which amounts to the sum of problems found to be theorems or counter-satisfiable. The columns WC and CPU refer to the average wall clock time and CPU time, respectively,

Semantics		**Leo-III**					**Avg. Time**		**U vs. Sat./Nit.**			**U vs. MleanCoP**		
System	Domains	Σ	THM	CSA	GUP	TMO	CPU	WC	Σ	THM	CSA	Σ	THM	CSA
D$_{sem}$	const$_{sem}$	195	195	0	57	314	21.2	9.2	14	14	0	5	5	0
D$_{sem}$	const$_{syn}$	174	174	†	19	373	31.0	12.2	13	13	†	5	5	†
D$_{syn}$	const$_{sem}$	194	194	0	0	372	20.8	9.1	14	14	0	5	5	0
D$_{syn}$	const$_{syn}$	171	171	†	0	395	30.5	12.2	13	13	†	5	5	†
D$_{sem}$	cumul$_{sem}$	178	178	†	19	369	21.4	9.4	13	13	†	5	5	†
D$_{sem}$	cumul$_{syn}$	174	174	†	19	373	29.2	11.6	13	13	†	5	5	†
D$_{syn}$	cumul$_{sem}$	178	178	†	0	388	22.4	9.8	13	13	†	5	5	†
D$_{syn}$	cumul$_{syn}$	171	171	†	0	395	28.7	11.6	13	13	†	5	5	†
D$_{sem}$	vary	161	161	†	69	336	22.5	10.0	13	13	†	5	5	†
D$_{syn}$	vary	161	161	†	0	405	22.9	10.1	13	13	†	5	5	†
T$_{sem}$	const$_{sem}$	246	246	0	54	266	22.1	9.1	17	17	0	6	6	0
T$_{sem}$	const$_{syn}$	218	218	†	22	326	29.4	11.3	14	14	†	6	6	†
T$_{syn}$	const$_{sem}$	235	235	0	0	331	28.2	11.0	22	22	0	6	6	0
T$_{syn}$	const$_{syn}$	209	209	†	0	357	40.7	14.7	20	20	†	6	6	†
T$_{sem}$	cumul$_{sem}$	230	230	†	22	314	22.8	9.6	15	15	†	6	6	†
T$_{sem}$	cumul$_{syn}$	219	219	†	22	325	25.9	10.4	15	15	†	6	6	†
T$_{syn}$	cumul$_{sem}$	211	211	†	0	355	25.8	10.3	18	18	†	6	6	†
T$_{syn}$	cumul$_{syn}$	210	210	†	0	356	35.4	13.3	19	19	†	6	6	†
T$_{sem}$	vary	211	211	†	65	290	22.5	9.6	15	15	†	6	6	†
T$_{syn}$	vary	193	193	†	0	373	27.1	10.9	18	18	†	6	6	†
S4$_{sem}$	const$_{sem}$	296	296	0	0	270	22.8	9.7	24	24	0	7	7	0
S4$_{sem}$	const$_{syn}$	245	245	†	0	321	28.0	11.4	17	17	†	7	7	†
S4$_{syn}$	const$_{sem}$	259	259	0	0	307	36.7	13.3	25	25	0	7	7	0
S4$_{syn}$	const$_{syn}$	227	227	†	0	339	44.6	15.4	19	19	†	7	7	†
S4$_{sem}$	cumul$_{sem}$	272	272	†	0	294	24.7	10.4	23	23	†	7	7	†
S4$_{sem}$	cumul$_{syn}$	253	253	†	0	313	28.3	11.5	18	18	†	7	7	†
S4$_{syn}$	cumul$_{sem}$	241	241	†	0	325	33.0	12.8	24	24	†	7	7	†
S4$_{syn}$	cumul$_{syn}$	235	235	†	0	331	38.5	14.5	22	22	†	7	7	†
S4$_{sem}$	vary	247	247	†	0	319	23.4	10.2	19	19	†	6	6	†
S4$_{syn}$	vary	223	223	†	0	343	35.2	13.5	21	21	†	6	6	†
S5$_{sem}$	const$_{sem}$	344	344	0	0	222	22.9	9.4	30	30	0	6	6	0
S5$_{sem}$	const$_{syn}$	285	285	†	0	281	29.1	11.6	19	19	†	6	6	†
S5$_{syn}$	const$_{sem}$	262	262	0	0	304	38.1	13.8	30	30	0	6	6	0
S5$_{syn}$	const$_{syn}$	233	233	†	0	333	48.6	17.1	26	26	†	6	6	†
S5$_{sem}$	cumul$_{sem}$	321	321	†	0	245	20.5	8.5	19	19	†	6	6	†
S5$_{sem}$	cumul$_{syn}$	283	283	†	0	283	26.3	10.3	19	19	†	6	6	†
S5$_{syn}$	cumul$_{sem}$	248	248	†	0	318	38.9	14.4	29	29	†	6	6	†
S5$_{syn}$	cumul$_{syn}$	237	237	†	0	329	44.3	15.9	27	27	†	6	6	†
S5$_{sem}$	vary	281	281	†	0	285	22.5	9.3	20	20	†	5	5	†
S5$_{syn}$	vary	225	225	†	0	341	38.6	14.3	25	25	†	5	5	†
S5U	const	440	440	0	45	81	26.2	11.2	71	71	0	15	15	0
S5U	cumul	439	439	0	45	82	25.6	11.0	70	70	0	15	15	0

Table 5: Results of Leo-III

of the total solutions. The number of problems with a specific encoding solved uniquely by either Leo-III or Satallax when compared to the other higher-order prover are broken down in *U vs. Sat.* and *U vs. Leo-III*, respectively, where again THM, CSA and Σ stand for found theorems, problems with found counter-models and the sum of both. Similarly, *U vs. MleanCoP* apportions the status THM, CSA and their sum Σ counting solutions uniquely found by the higher-order prover when compared to the native prover MleanCoP. For any field that counts the number of problems with status CSA,

Semantics		**Satallax**					**Avg. Time**		**U vs. Leo-III/Nit.**			**U vs. MleanCoP**		
System	Domains	Σ	THM	CSA	GUP	TMO	CPU	WC	Σ	THM	CSA	Σ	THM	CSA
$\mathrm{D_{sem}}$	$\mathrm{const_{sem}}$	189	183	6	0	377	30.4	24.3	2	2	0	1	1	0
$\mathrm{D_{sem}}$	$\mathrm{const_{syn}}$	178	177	†	0	388	32.1	25.1	16	16	†	1	1	†
$\mathrm{D_{syn}}$	$\mathrm{const_{sem}}$	183	183	0	0	383	38.1	31.9	3	3	0	1	1	0
$\mathrm{D_{syn}}$	$\mathrm{const_{syn}}$	177	177	†	0	389	38.4	32.0	19	19	†	1	1	†
$\mathrm{D_{sem}}$	$\mathrm{cumul_{sem}}$	168	168	†	0	397	29.0	22.6	3	3	†	1	1	†
$\mathrm{D_{sem}}$	$\mathrm{cumul_{syn}}$	168	168	†	0	397	30.8	24.3	7	7	†	1	1	†
$\mathrm{D_{syn}}$	$\mathrm{cumul_{sem}}$	168	168	†	0	398	36.8	30.5	3	3	†	1	1	†
$\mathrm{D_{syn}}$	$\mathrm{cumul_{syn}}$	168	168	†	0	398	38.6	32.0	10	10	†	1	1	†
$\mathrm{D_{sem}}$	vary	151	151	†	0	411	26.7	19.9	3	3	†	1	1	†
$\mathrm{D_{syn}}$	vary	151	151	†	0	415	35.4	28.8	3	3	†	1	1	†
$\mathrm{T_{sem}}$	$\mathrm{const_{sem}}$	297	236	61	0	269	26.0	20.8	7	7	0	14	3	11
$\mathrm{T_{sem}}$	$\mathrm{const_{syn}}$	255	226	†	0	311	33.3	27.4	22	22	†	5	5	†
$\mathrm{T_{syn}}$	$\mathrm{const_{sem}}$	226	226	0	0	340	41.3	35.2	13	13	0	3	3	0
$\mathrm{T_{syn}}$	$\mathrm{const_{syn}}$	219	219	†	0	347	42.1	35.4	30	30	†	5	5	†
$\mathrm{T_{sem}}$	$\mathrm{cumul_{sem}}$	223	223	†	0	279	28.3	22.4	8	8	†	5	5	†
$\mathrm{T_{sem}}$	$\mathrm{cumul_{syn}}$	221	221	†	0	321	33.5	27.3	17	17	†	5	5	†
$\mathrm{T_{syn}}$	$\mathrm{cumul_{sem}}$	215	215	†	0	351	43.0	36.6	22	22	†	5	5	†
$\mathrm{T_{syn}}$	$\mathrm{cumul_{syn}}$	212	212	†	0	354	43.4	36.8	21	21	†	5	5	†
$\mathrm{T_{sem}}$	vary	203	203	†	0	287	26.8	20.5	7	7	†	5	5	†
$\mathrm{T_{syn}}$	vary	196	196	†	0	370	41.0	34.4	21	21	†	5	5	†
$\mathrm{S4_{sem}}$	$\mathrm{const_{sem}}$	330	279	51	0	236	31.0	25.6	7	7	0	13	4	9
$\mathrm{S4_{sem}}$	$\mathrm{const_{syn}}$	289	263	†	0	277	36.9	30.9	35	35	†	4	4	†
$\mathrm{S4_{syn}}$	$\mathrm{const_{sem}}$	261	261	0	0	305	53.6	47.6	27	27	0	4	4	0
$\mathrm{S4_{syn}}$	$\mathrm{const_{syn}}$	256	256	†	0	310	47.8	41.1	48	48	†	4	4	†
$\mathrm{S4_{sem}}$	$\mathrm{cumul_{sem}}$	263	263	†	0	249	31.7	25.8	14	14	†	4	4	†
$\mathrm{S4_{sem}}$	$\mathrm{cumul_{syn}}$	261	261	†	0	284	38.2	31.9	26	26	†	4	4	†
$\mathrm{S4_{syn}}$	$\mathrm{cumul_{sem}}$	251	251	†	0	315	48.5	42.5	34	34	†	4	4	†
$\mathrm{S4_{syn}}$	$\mathrm{cumul_{syn}}$	248	248	†	0	318	49.7	42.9	35	35	†	4	4	†
$\mathrm{S4_{sem}}$	vary	239	239	†	0	261	31.3	25.2	11	11	†	2	2	†
$\mathrm{S4_{syn}}$	vary	228	228	†	0	338	46.1	40.0	26	26	†	2	2	†
$\mathrm{S5_{sem}}$	$\mathrm{const_{sem}}$	356	324	32	0	210	31.8	26.3	10	10	0	11	3	8
$\mathrm{S5_{sem}}$	$\mathrm{const_{syn}}$	324	310	†	0	242	39.6	33.6	44	44	†	3	3	†
$\mathrm{S5_{syn}}$	$\mathrm{const_{sem}}$	270	270	0	0	296	52.9	47.5	38	38	0	3	3	0
$\mathrm{S5_{syn}}$	$\mathrm{const_{syn}}$	254	254	†	0	312	50.1	44.4	47	47	†	3	3	†
$\mathrm{S5_{sem}}$	$\mathrm{cumul_{sem}}$	324	324	†	0	217	32.7	27.0	22	22	†	3	3	†
$\mathrm{S5_{sem}}$	$\mathrm{cumul_{syn}}$	312	312	†	0	245	40.8	34.8	48	48	†	3	3	†
$\mathrm{S5_{syn}}$	$\mathrm{cumul_{sem}}$	256	256	†	0	310	52.1	46.6	37	37	†	3	3	†
$\mathrm{S5_{syn}}$	$\mathrm{cumul_{syn}}$	248	248	†	0	318	51.8	45.8	38	38	†	3	3	†
$\mathrm{S5_{sem}}$	vary	280	280	†	0	235	32.8	26.6	19	19	†	1	1	†
$\mathrm{S5_{syn}}$	vary	237	237	†	0	329	51.3	45.4	37	37	†	1	1	†
S5U	const	410	369	41	0	156	27.5	21.1	0	0	0	20	7	13
S5U	cumul	410	369	41	0	156	27.1	21.2	0	0	0	20	7	13

Table 6: Results of Satallax

the symbol † indicates that the encoding does not allow for finding counter-models due to the incompleteness issue with varying quantifiers in the encoding.

When it comes to theorem proving, encoding modal systems X where X ∈ {D,T,S4} with properties on the accessibility relation ($\mathrm{X_{sem}}$) and quantification with varying quantifier for varying domains, domain restrictions in the case of cumulative domains ($\mathrm{cumul_{sym}}$), and the special const-quantifier in case of constant domains ($\mathrm{const_{sym}}$) clearly yields the best results for both theorem provers Leo-III and Satallax. This encoding outperforms

Semantics		**Nitpick**					**Avg. Time**		**U vs. Leo-III/Sat.**			**U vs. MleanCoP**		
System	Domains	Σ	THM	CSA	GUP	TMO	CPU	WC	Σ	THM	CSA	Σ	THM	CSA
D_{sem}	$const_{sem}$	294	0	294	39	230	45.4	27.8	288	0	288	102	0	102
D_{syn}	$const_{sem}$	294	0	294	43	226	45.2	27.1	294	0	294	102	0	102
T_{sem}	$const_{sem}$	182	0	182	53	330	45.3	27.6	121	0	121	76	0	76
T_{syn}	$const_{sem}$	184	0	184	63	317	46.1	28.4	184	0	184	78	0	78
$S4_{sem}$	$const_{sem}$	123	0	123	4	437	40.4	23.4	72	0	72	48	0	48
$S4_{syn}$	$const_{sem}$	123	0	123	81	360	40.0	24.0	123	0	123	48	0	48
$S5_{sem}$	$const_{sem}$	68	0	68	4	492	40.3	23.6	36	0	36	35	0	35
$S5_{syn}$	$const_{sem}$	68	0	68	107	389	41.7	23.9	68	0	68	35	0	35
S5U	const	73	0	73	157	334	41.3	24.5	32	0	32	40	0	40
S5U	cumul	73	0	73	157	334	40.9	23.9	32	0	32	40	0	40

Table 7: Results of Nitpick

Semantics		**MleanCoP**					**Avg. Time**		**U vs. HOany**		
System	Domains	Σ	THM	CSA	GUP	TMO	CPU	WC	Σ	THM	CSA
D	const	430	218	212	0	136	5.1	5.7	46	26	20
D	cumul	439	203	236	0	127	5.8	6.6	263	27	236
D	vary	444	185	259	0	122	6.3	7.1	285	26	259
T	const	372	266	106	0	194	2.9	3.8	20	20	0
T	cumul	370	247	123	0	196	3.5	4.3	141	18	123
T	vary	369	223	146	0	197	4.4	5.2	160	14	146
S4	const	436	361	75	0	130	8.5	9.1	66	66	0
S4	cumul	433	346	87	0	133	8.0	8.7	155	68	87
S4	vary	403	288	115	0	163	8.2	8.8	151	36	115
S5	const	464	431	33	0	102	4.5	5.2	6	6	0
S5	cumul	464	431	33	0	102	4.5	5.3	7	7	0
S5	vary	441	359	82	0	125	4.7	5.5	146	64	82

Table 8: Results of MleanCoP

any other encoding when combined with Leo-III by 12.1%, 2.3%, 0.0% minimally and 30.4%, 15.7%, 10.8% maximally, and when paired with Satallax by 3.4%, 0.0%, 0.0% minimally and 9.0%, 7.8%, 4.8% maximally, for constant, cumulative and varying quantification semantics, respectively. For modal system S5 this statement also holds for varying quantifiers, too, $S5_{sem}$ being 24.9% stronger than $S5_{syn}$. If encoded with modal system S5U instead of S5, constant and cumulative quantification semantics, respectively, receive a significant boost of 27.9%, 37.1% minimally and 88.8%, 85.7% maximally when using Leo-III and 13.9%, 13.9% minimally and 45.3%, 48.8% maximally when using Satallax.
The modal system encoding does not seem to have any effect when searching for counter-models for constant domain semantics with Nitpick since the numbers are identical except in the case of modal system T for which T_{syn} exposes two (1.1%) more problems as counter-satisfiable than T_{sem} and of course in the case of modal system S5. Nitpick finds finds five (7.4%) more solutions with S5U than with $S5_{syn}$ or $S5_{sem}$. In addition, S5U also permits finding valid counter-models for cumulative domain semantics.
This results suggests that encodings which encourage blind instantiation in the proving process like modality axiom schemes and Barcan or converse Barcan formula are severely

Semantics		**Leo-III**				**Satallax**				**HOany**	**MleanCoP**
System	Domains	sem sem	sem syn	syn sem	syn syn	sem sem	sem syn	syn sem	syn syn		
D	const	195	174	194	171	183	177	183	177	197	218
D	cumul	178	174	178	171	168	168	168	168	181	203
D	vary	161	-	161	-	151	-	151	-	164	185
T	const	246	218	235	209	236	226	226	219	253	266
T	cumul	230	219	211	210	223	221	215	212	238	247
T	vary	211	-	193	-	203	-	196	-	218	223
S4	const	296	245	259	227	279	263	261	256	303	361
S4	cumul	272	253	241	235	263	261	251	248	286	346
S4	vary	247	-	223	-	239	-	228	-	258	288
S5	const	440	-	-	-	369	-	-	-	440	431
S5	cumul	439	-	-	-	369	-	-	-	439	431
S5	vary	281	-	225	-	280	-	237	-	300	359

Table 9: Comparison of provers for found theorems

weaker than encodings that state meta-characteristics about the logic e.g. properties imposed upon the accessibility relation and structure of the domains.
For the remainder of this evaluation the best encoding i.e. the encoding that yields most theorems (and almost always most problems found counter-satisfiable) as described in this paragraph is assumed if the encoding is not specified otherwise.

The average reasoning time for the best encoding for any prover is always better except for Leo-III in semantics modal system D and constant domains (0.1s/0.4s or 1.1%/1.9% slower) or varying domains (0.1s/0.4s or 1.0%/1.8% slower) as well as modal system S5 for constant domains (1.8s/3.3s or 5.9%/14.4% slower) or cumulative domains (3.5s/5.1s or 29.4%/24.9% slower) for WC/CPU time, respectively, when compared to the fastest encoding. A further exception is Nitpick in modal system D with constant quantification semantics (0.7/0.2s or 2.6%/0.4% slower) or modal system S4 with constant quantification semantics (WC 0.4s or 1.0% slower, CPU is faster) for WC/CPU time, respectively, when compared to the fastest encoding. The best encoding has 49.6% faster WC and 39.0% faster CPU time on average compared to all other encodings.
In conclusion the best encoding in terms of found theorems is usually the fastest encoding if WC and CPU time is of concern with an exception of a marginal surplus in WC and CPU time for three different semantics and a medium increase for two other semantics.

Comparison of solved theorems by the MET and MleanCoP. Table 8 accounts for each semantics (modal system and domains) the individual performance in terms of the SZS status theorem (THM), counter-satisfiablity (CSA), gave-up (GUP), timeout (TMO), the combined solutions of THM and CSA as Σ, the average CPU time (CPU) and wall clock time (WC). *U vs.* HOany counts only those problems with status THM, CSA and both (Σ) that could not be solved by any higher-order prover with the best encoding.

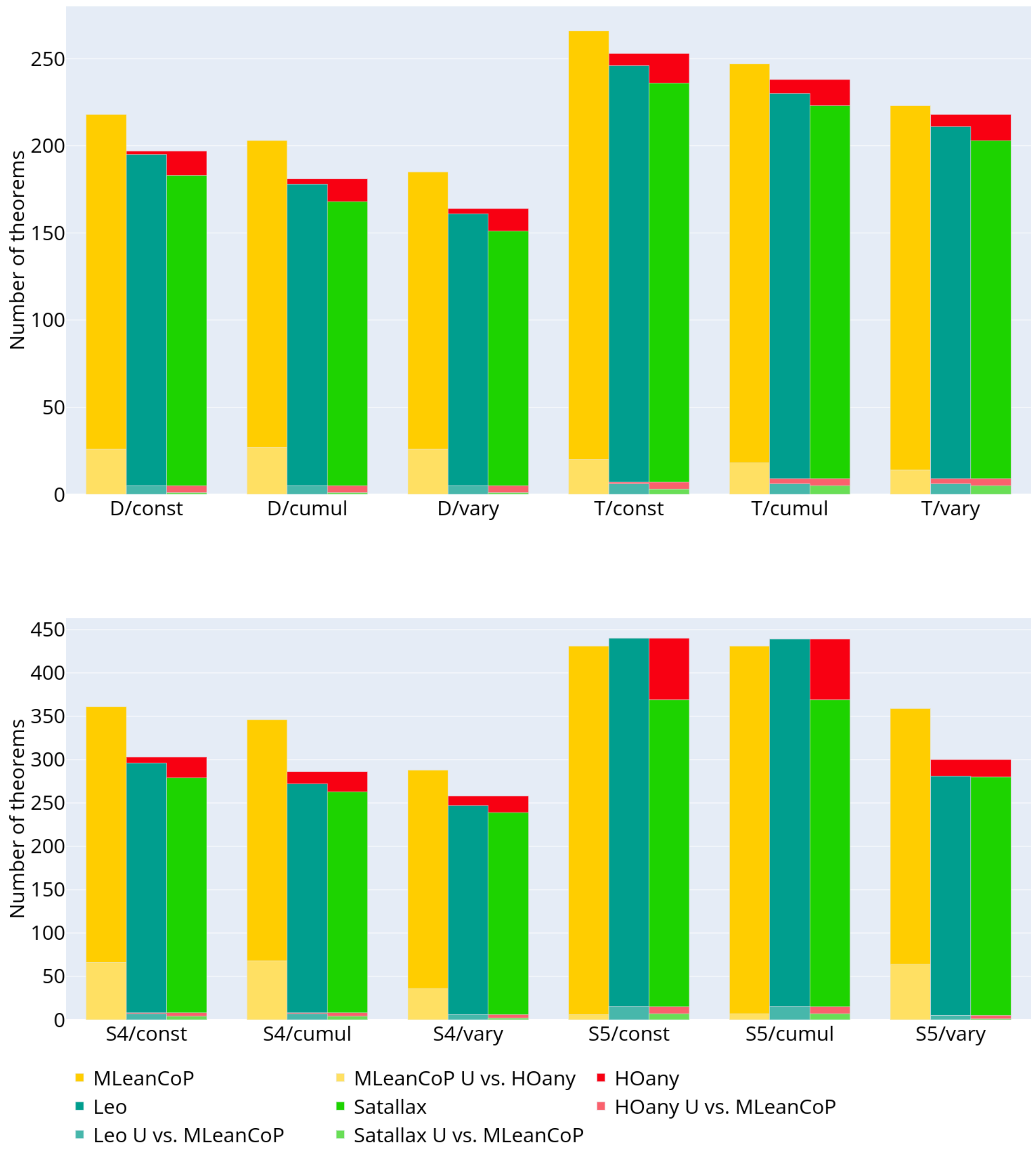

Figure 3: Comparison of provers for found theorems

Table 9 compares the performance in terms of found theorems for all theorem provers. For each semantics i.e. modal system and quantification semantics, for both Leo-III and Satallax the number of found theorems is counted, once for each possible encoding of the current semantics. The column of one encoding is titled with a tuple from the set $\{\text{sem,syn}\}^2$ for which the first item of the tuple represents the encoding of the modal

system and the second item the quantification encoding using the same terminology as described in Tables 4a and 4b. In the case of modal system S5, the optimized encoding S5U is applied for constant and cumulative quantification semantics. The hyphen marks encodings that are not applicable e.g. there is only one encoding for varying quantifiers or modal system S5U which does not possess an explicit modal operator's axiomatization. Column HOAny counts the union of theorems found by Leo-III or Satallax for the best encoding. Column MleanCoP accounts for number of theorems found by MleanCoP.
Figure 3 illustrates the number of theorems found by all theorem provers (vertical axis) for different semantics (plot sub-captions). Bars with dark tonalites of yellow, turqouise, green and red count theorems found by MleanCoP, Leo-III, Satallax and HOAny, respectively, where only the best encoding is employed and HOAny again is the union of results for the best encoding when employing Leo-III or Satallax. Light tonalities count uniquely found theorems by a prover against the reference system: Light yellow considers only those theorems found by MleanCoP that could not be solved by any higher-order prover with the best encoding or in other words found by HOAny. Light turquoise, light green and light red count theorems uniquely found by Leo-III, Satallax and HOAny (i.e. by Leo-III or Satallax for the best encoding), respectively, when compared against MleanCoP.

For any semantics and any encoding Leo-III solves more theorems than Satallax. In total this amounts to 7.9% more theorems for the best encoding.
Across all semantics, Leo-III was able to uniquely prove 314 theorems compared to Satallax and Satallax could uniquely prove 81 theorems compared to Leo-III. As model finder Nitpick does not prove any theorems.
Therefore Leo-III is the obvious choice if only one theorem prover paired with the MET is consulted.

In total, MleanCoP solved 3558 theorems, Leo-III 3196 (MleanCoP 11.3% stronger), Satallax 2963 (MleanCoP 20.1% stronger) and HOAny 3277 (MleanCoP 8.6% stronger). The encoding S5U closes the gap between the MET and MleanCoP for modal system S5 in which MleanCoP solved 431/431 (HOAny 2.3%/2.1% stronger) theorems, Leo-III 440/439 (HOAny 0.0%/0.0% stronger), Satallax 369/369 (HOAny 19.2%/19.0% stronger) and HOAny 440/439 for constant/cumulative quantification semantics, respectively. For all other semantics MleanCoP is stronger compared to Leo-III by 8.1%, 7.4%, 5.7% minimally, 14.7%, 17.1%, 17.2% on average, and 22.0%, 27.2%, 27.8% maximally, compared to Satallax by 12.7%, 10.8%, 9.9% minimally, 21.1%, 21.7%, 20.8% on average, and 29.4%, 31.6%, 28.2% maximally, and, compared to HOAny by 5.1%, 3.8%, 2.3% minimally, 12.2%, 12.9%, 12.2% on average, and 19.1%, 21.0%, 19.7% maximally, for constant, cumulative and varying quantification semantics, respectively.
For any semantics, Leo-III was capable of successfully proving five to 15 theorems that could not be discovered by MleanCoP, as for Satallax this number ranges between one and seven.

Semantics		**Nitpick**		**MleanCoP**
System	Domains	sem sem	syn sem	
D	const	294	294	212
T	const	182	184	106
S4	const	123	123	75
S5	const	73	-	33
S5	cumul	73	-	33

Table 10: Comparison of provers for found counter-models

The amount of solved theorems by Leo-III or Satallax compared to MleanCoP suggests the semantic embedding approach for modal logics is indeed a fruitful and competetive approach. It is even more powerful when more than one higher-order prover is employed in parallel as shown in the case of HOANY, and probably more so if they operate on different calculus paradigms (tableaux, paramodulation, superposition, etc.). Since there exist a fair number higher-order provers, the strength of HOANY is expected to increase further if the MET is coupled with more reasoning systems. Further evidence provides the amount of unique solutions found by higher-order provers that could not be successfully proven by the strongest native prover MleanCoP.

Comparison of problems found counter-satisfiable by the MET and MleanCoP. Table 10 compares the performance of MleanCoP and Nitpick in terms of problems that have been identified as counter-satisfiable. For each semantics i.e. modal system and quantification semantics that is available for finding counter-models when using the MET (i.e. that does not suffer from the incompleteness issue), the number of problems found counter-satisfiable by Nitpick are listed for each encoding that ensures only valid counter-model can be discovered. The column of one encoding is titled with a tuple from the set {(sem,sem),(syn,sem)} for which the first item of the tuple represents the encoding of the modal system and the second item the quantification encoding using the same terminology as described in Tables 4a and 4b. In the case of modal system S5, the optimized encoding S5U is applied for constant and cumulative quantification semantics. Column MleanCoP accounts for the number of problems found counter-satisfiable by MleanCoP.
Figure 4 of problems identified as counter-satisfiable by MleanCoP and Nitpick for different semantics (plot sub-captions). Bars with dark tonalites of yellow and blue count problems for which counter-models were found by MleanCoP and Nitpick, respectively. Light tonalities indicate problems for which the corresponding reference reasoning system could not find any counter-model i.e. light yellow counts problems that were uniquely identified as counter-satisfiable by MleanCoP when compared to Nitpick and light blue vice versa.

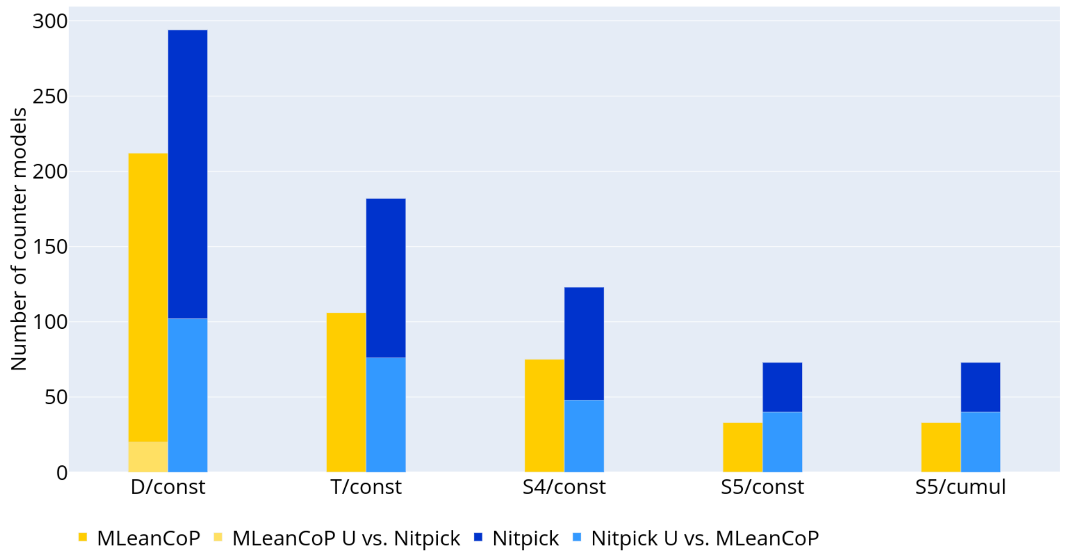

Figure 4: Comparison of provers for found counter-models

As for finding counter-models, Leo-III was not able to find any and all problems found counter-satisfiable by Satallax are already successfully solved by Nitpick. Also Nitpick is 272.5% stronger than Satallax in fulfilling this task hence there are no advantages in choosing Satallax over Nitpick when proving problems to be counter-satisfiable.
This outcome for finding counter-models and theorem proving (see last paragraph 5.2) once again provides evidence that it is beneficial to make use of various reasoning systems in parallel, since they possess different strengths, either due to their underlying calculus which is based on paramodulation for Leo-III and tableaux for Satallax, or due to their inherent nature such as being a theorem prover (Leo-III), mixed system (Satallax) or model finder (Nitpick).

Nitpick finds counter-models for 745 problems while MleanCoP finds only 459 which makes Nitpick 62.3% stronger than MleanCoP when finding problems to be counter-satisfiable but only for the supported semantics of this task. Nitpick preveils with 38.7% minimally and 121.2% maximally more problems found counter-satisfiable on these semantics when compared to MleanCoP. Uniquely solved problems by MleanCoP appear only regarding modal system D (20 such problems) when compared to Nitpick, which itself finds 102 problems counter-satisfiable that MleanCoP cannot solve in modal system D. For modal systems T,S4,S5 the number of unique solutions produced by Nitpick ranges between 40 and 76.

Semantics		**Leo-III**				**Satallax**				**MleanCoP**
System	Domains	sem sem	sem syn	syn sem	syn syn	sem sem	sem syn	syn sem	syn syn	
D	const	9.2	12.2	9.1	12.2	23.5	25.2	31.9	32.0	6.9
D	cumul	9.4	11.6	9.8	11.6	22.6	24.3	30.5	32.0	7.2
D	vary	10.0	-	10.1	-	19.9	-	28.8	-	8.4
T	const	9.1	11.3	11.0	14.7	20.9	27.9	35.2	35.4	4.0
T	cumul	9.6	10.4	10.3	13.3	22.4	27.3	36.6	36.8	3.7
T	vary	9.6	-	10.9	-	20.5	-	34.4	-	5.0
S4	const	9.7	11.4	13.3	15.4	25.8	31.4	47.6	41.1	10.1
S4	cumul	10.4	11.5	12.8	14.5	25.8	31.9	42.5	42.9	9.9
S4	vary	10.2	-	13.5	-	25.2	-	40.0	-	10.3
S5	const	11.2	-	-	-	21.2	-	-	-	5.2
S5	cumul	11.0	-	-	-	21.2	-	-	-	5.3
S5	vary	9.3	-	14.3	-	26.6	-	45.4	-	5.8

Table 11: Comparison of provers for average WC when finding theorems

Semantics		**Leo-III**				**Satallax**				**MleanCoP**
System	Domains	sem sem	sem syn	syn sem	syn syn	sem sem	sem syn	syn sem	syn syn	
D	const	21.2	31.0	20.8	30.5	29.7	32.2	38.1	38.4	6.4
D	cumul	21.4	29.2	22.4	28.7	29.0	30.8	36.8	38.6	6.5
D	vary	22.5	-	22.9	-	26.7	-	35.4	-	7.7
T	const	22.1	29.4	28.2	40.7	26.7	34.1	41.3	42.1	3.2
T	cumul	22.8	25.9	25.8	35.4	28.3	33.5	43.0	43.4	3.0
T	vary	22.5	-	27.1	-	26.8	-	41.0	-	4.2
S4	const	22.8	28.0	36.7	44.6	31.6	37.7	53.6	47.8	9.5
S4	cumul	24.7	28.3	33.0	38.5	31.7	38.2	48.5	49.7	9.2
S4	vary	23.4	-	35.2	-	31.3	-	46.1	-	9.7
S5	const	26.2	-	-	-	27.9	-	-	-	4.5
S5	cumul	25.6	-	-	-	27.5	-	-	-	4.5
S5	vary	22.5	-	38.6	-	32.8	-	51.3	-	5.0

Table 12: Comparison of provers for average CPU when finding theorems

These results show that MET+Nitpick possesses a considerable advantage over the native prover MleanCoP. This large gap also suggests once the incompleteness issue is fixed, the MET might be better suited at this task than MleanCoP for other semantics, too. Furthermore this result provides evidence that splitting theorem proving and counter-model finding into two different reasoning processes or tools that specialize in the regarding reasoning task, increases the number of total solutions. Hence building a native reasoner that excels in both tasks might involve even more expense than pictured in paragraph 5.2.

Comparison of CPU and WC usage by the MET and MleanCoP. Tables 11 and 12 display the average wall clock time and CPU time, respectively, of Leo-III, Satallax and MleanCoP when successfully producing the status theorem for all semantics available to MleanCoP. Analogously, Tables 13 and 14 show the average wall clock

Semantics		**Nitpick**		**MleanCoP**
System	Domains	sem sem	syn sem	
D	const	27.8	27.1	4.6
T	const	27.6	28.4	3.2
S4	const	23.4	24.0	4.3
S5	const	24.5	-	5.4
S5	cumul	23.9	-	5.4

Table 13: Comparison of provers for average WC when finding counter-models

Semantics		**Nitpick**		**MleanCoP**
System	Domains	sem sem	syn sem	
D	const	45.4	45.2	3.7
T	const	45.3	46.1	2.3
S4	const	40.4	40.0	3.4
S5	const	41.3	-	4.4
S5	cumul	40.9	-	4.5

Table 14: Comparison of provers for average CPU when finding counter-models

time and CPU time, respectively, of Nitpick and MleanCoP when successfully finding a counter-model for eligible semantics i.e. semantics that are not affected by the incompleteness issue. For each semantics (modal system and quantification semantics) and all higher-order provers, the average reasoning time of any applicable encoding of the semantics is calculated. The column of one encoding is titled with a tuple from the set $\{\text{sem,syn}\}^2$ for which the first item of the tuple represents the encoding of the modal system and the second item the quantification encoding using the same terminology as described in Tables 4a and 4b. In the case of modal system S5, the optimized encoding S5U is applied for constant and cumulative quantification semantics. Encodings that are not supported for the regarding task are marked with a hypen or do not possess a column of their own.
Figure 5 and 6 illustrate the reasoning times (vertical axis) for eligible semantics with respect to the reasoning task (plot sub-captions) on different reasoning systems for theorem proving and counter-model finding, respectively. Dark tonalities indicate wall clock time, light tonalities show CPU time for the reasoning systems MleanCoP, Leo-III, Satallax and Nitpick that are represented by the colors colors yellow, turquoise, green and blue, respectively. CPU and WC for MleanCoP is almost identical and therefore the plot of CPU covers most of the plot of WC.

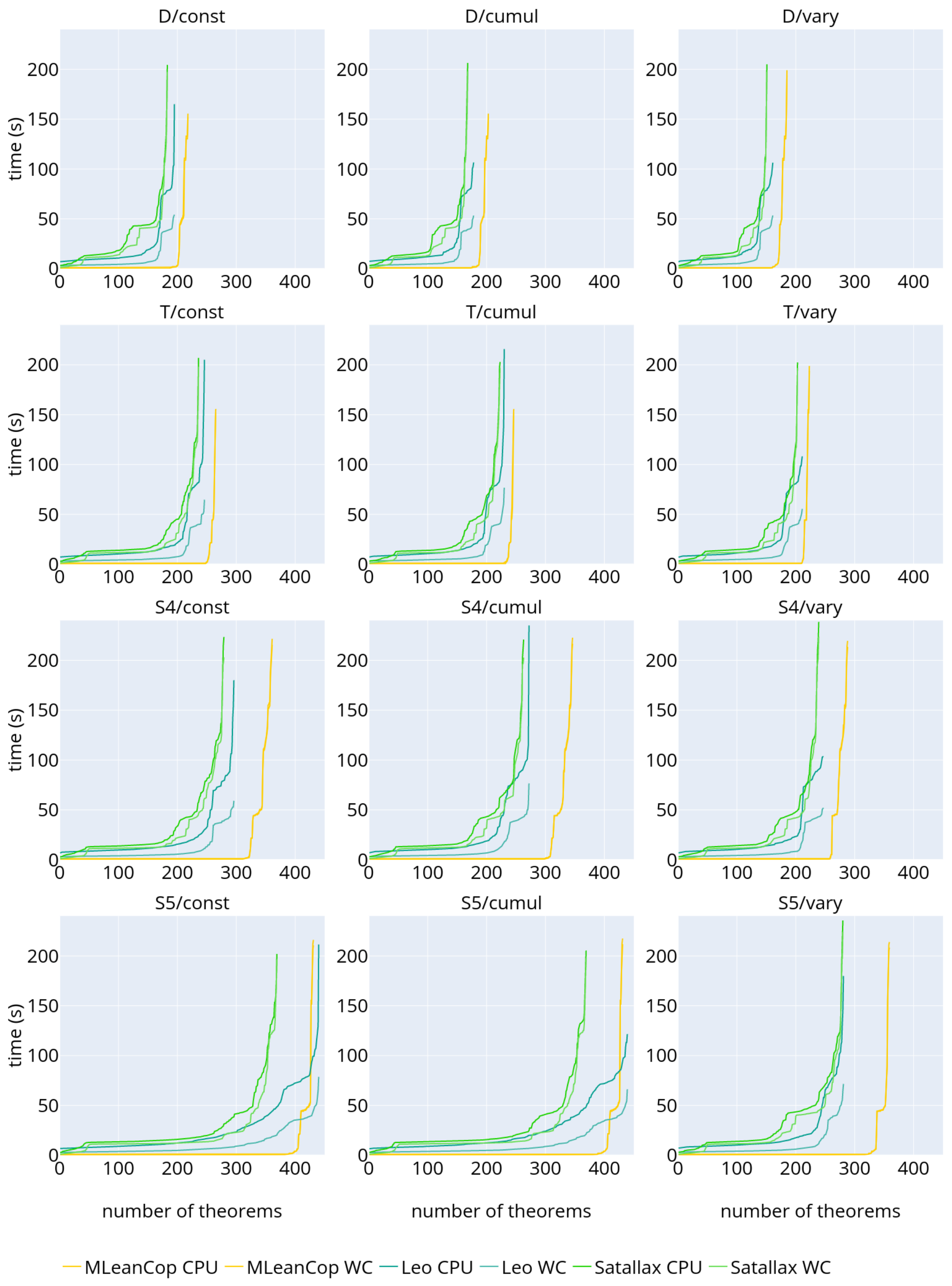

Figure 5: Comparison of provers for average WC and CPU when finding theorems

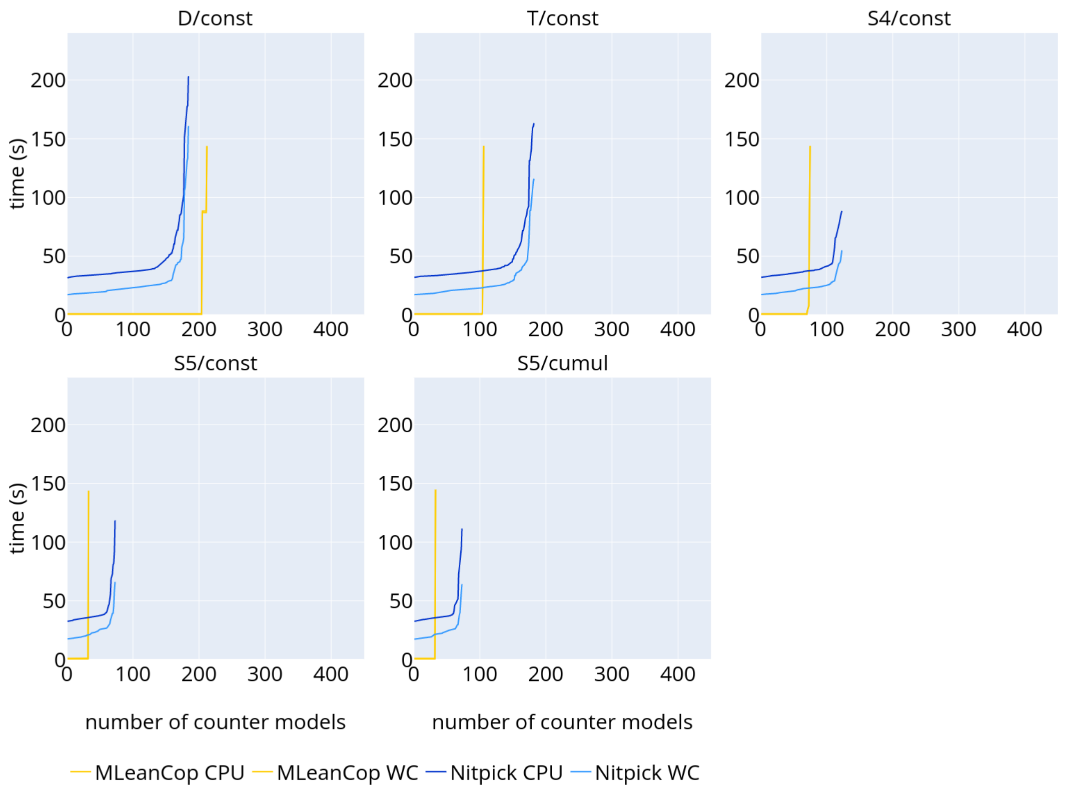

Figure 6: Comparison of provers for average WC and CPU when finding counter-models

For the best encoding Leo-III takes on average 61.9%/339.4% more time than Mlean-CoP, maximally 126.8%/647.6% for WC/CPU time, respectively, across all semantics. In modal system S4 Leo-III is able to outpace MleanCoP in terms of WC time by 4.1% for constant and 1.0% for varying domain semantics but is 5.1% slower in the cumulative case. For the best encoding Satallax takes on average 271.3%/444.9% more time than MleanCoP, minimally 137.0%/220.9% maximally 502.0%/831.1% for WC/CPU time, respectively, across all semantics.
Figure 5 illustrates that the time needed in the proving process is under five seconds for almost all problems solved by MleanCoP. After roughly 100 seconds, there is mostly no improvement in terms of found theorems for Leo-III and after 130 seconds for Satallax. In the case of counter-model finding, Nitpick is substantially slower than MleanCoP, on average 483.0%/1147.1%, minimally 343.3%/815.2% and maximally 761.4%/1882.9%, for WC/CPU time, respectively.
If speed or CPU usage is of the essence, the native prover MleanCoP will be significantly ahead compared to the MET paired with higher-order provers, regardless of the reasoning task.

Summary. In this evaluation the following results have been found:

- No errors or unexpected results occurred employing any software mentioned in the evaluation.
- The expense of implementing the semantic embedding approach is substantially less than devising a specialized reasoning system.
- Encodings utilizing varying quantifiers have been experimentally proven to be incomplete.
- The MET supports a much wider range of semantics than the native prover MleanCoP.
- Encodings that expose semantics more explicitly lead to more solutions than those encodings that apply axiom schemes.
- The number of theorems found when using the MET is less than employing MleanCoP but still competitive.
- The number of problems identified as counter-satisfiable for eligible semantics of the MET is considerable higher for the MET+Nitpick when compared to MleanCoP.
- Running multiple reasoning systems in parallel (paired with the MET) is a sensible approach, yielding more solutions overall.
- MleanCoP is significantly faster than any combination of the MET and a higher-order reasoner.

6 Conclusion and Outlook

The first part of this thesis presented a general version of higher-order modal logics based on the simply typed lambda calculus. An extension of the TPTP THF dialect addressing the representation of modal logic problems and their semantics was devised, which is planned to be included into the TPTP framework.
Furthermore, the semantic embedding approach was demonstrated for higher-order modal logics. Its implementation in the software MET has been further developed and now yields a transformation procedure enabling the automation of higher-order modal logics for problems formulated in the afore mentioned input format. An extensive evaluation of the different encodings and their comparison to the strongest reasoner for quantified modal logics has been conducted. The main findings are as follows:

- Semantic embedding is an effective shortcut for automating non-classical logics while at the same time allowing an easy adaption for a vast amount of variations on a specific non-classical logic.
- Theorem proving employing the MET is, despite some shortcomings, competitive with respect to the amount of theorems found. Also, the approach will automatically improve over time as higher-order ATP systems are constantly being improved.
- Counter-model finding employing the MET is highly competitive with respect to the number of problems identified as counter-satisfiable.
- Pairing the MET with higher-order reasoners is substantially slower than using a native prover. However, when speed is not of major concern, this negative aspect seems merely a minor disadvantage.
- Parallel deployment of multiple reasoning systems to one reasoning task is beneficial in the case of the semantic embedding of higher-order modal logics, allowing different provers

In conclusion, the semantic embedding approach for higher-order modal logics is very promising if the focus lies on finding solutions. Since encodings of other non-classical logics are mostly equally or less complex effective automation via semantic embedding may also apply to further application domains.

Further Work

The theoretical aspects of embedding higher-order modal logics leaves much work to do. First of all, the incompleteness issue has to be addressed which will render the embedding useful for counter-model finding for quantification semantics other than constant domains. This not only concerns varying quantifiers and domain restrictions but also the validity of CBF and BF in the higher-order context which is of interest in the actualism vs. possibilism discussion. Secondly, a soundness and completeness proof for the

encoding should be pursued to ensure an appropriate mapping of higher-order modal logics to HOL and provide trust to the users of the MET. A matter not discussed in this thesis is the encoding of equality. Its meaning in Kripke models is certainly ambiguous and employing native HOL equality is probably challenging. A feasible solution could be an approach that axiomatizes (different kinds of) equality. Last but not least the conjectured correspondence of first-order models between first-order modal logics [23, 22] and higher-order modal logics should be formally assessed and verified in order to assure higher-order modal logics as characterized in this thesis actually is a conservative extension of first-order modal logics. Furthermore, the enforcement of domain restrictions on all accessibility relations for cumulative and decreasing quantification semantics should be studied and possible alternatives should be explored, especially if different modal operators interact with each other on formula level. Exposing the differences of local and global consequence relations with any mix of local and global assumptions with respect to the implications in both theoretical and practical applications certainly is worth examining, too.
The implementation of the MET is still ongoing. Current and further work includes supporting flexible constants as well as cumulative and decreasing quantification semantics in multi-modal environments. Moreover, utilizing polymorphic quantifiers, which are part of the THF format would permit a more dense encoding that does not possess separated quantifier definitions or axioms for each type. Employing polymorphism seems to yield better reasoning results [20]. However, this option has neither been realized nor examined yet for encoding higher-order modal logic. Furthermore, once equalities have been investigated properly with respect to the shallow semantic embedding approach, supporting their encoding will provide a significant advantage for the MET due to the very low count of reasoning tools for modal logics with equality (cf. [43]). Finally, a more general version of higher-order modal logics might allow separate sets of worlds, one for each accessibility relation (cf. [18]). The adaptation is an easy one and if implemented it would render the MET even more versatile than it already is and once more convict the semantic embedding approach being a fruitful and reasonable undertaking for automating non-classical logics.

A List of Figures

B List of Tables

C List of Definitions

D CLI Parameters

Option	Eligible Values for `<arg>`	Effect
`-i <arg>`	existing file or directory	specify input file or directory
`-o <arg>`	file if input is a file, directory if input is a directory	specify output file or directory
`-diroutput <arg>`	joint, splitted	necessary if the input is a directory; `joint` mirrors the input (sub-) directories, multiple embeddings per problem are created with the semantics encoded in the filename of each resulting problem; `splitted` mirrors the input (sub-)directories, one for each specified semantics
`-constants <arg>`	rigid, flexible	provide default constants semantics if the problem does not include a logic specification, flexible not yet supported
`-domains <arg>`	constant, cumulative, decreasing, varying	provide default quantification semantics
`-consequences <arg>`	local, global	provide default consequence semantics
`-systems <arg>`	K, KB, K4, K5, K45, KB5, KB5_KB5, KB5_KB4, KB5_KB45, D, DB, D4, D5, D45, T, B, S4, S5, S5_KT5, S5_KTB5, S5_KT45, S5_KTB4, S5_KDB4, S5_KDB45, S5_KDB5, S5U	provide default modal operators semantics
`-t <arg>`	semantic_decreasing_quantification, semantic_constant_quantification, syntactic_modality_axiomatization, syntactic_constant_quantification, syntactic_cumulative_quantification, semantic_cumulative_quantification, semantic_modality_axiomatization, syntactic_decreasing_quantification	transformation parameters specifying parts of the encoding; syntactic means employing axiom schemes; everything else has prefix semantic
`-log <arg>`	file	save log to the specified file
`-h`	n/a	displays help

Exemplary usage

Embed problem `my_input_problem.p` which contains a logic specification:

```
java -jar embed-1.0-SNAPSHOT-shaded.jar
    -i my_input_problem.p
    -o my_output_problem.p
```

Embed problem `my_input_problem.p` which possesses no logic specification and provide the intended semantics via CLI:

```
java -jar embed-1.0-SNAPSHOT-shaded.jar
    -i my_input_problem.p
    -o my_output_problem.p
    -constants rigid
    -domains cumulative
    -consequences global
    -systems S5
```

Embed problem `my_input_problem.p` which possesses no logic specification and provide multiple semantics (the cross product of all supplied semantics options) via CLI, resulting in multiple files containing different embeddings written to directory `my_output_dir`:

```
java -jar embed-1.0-SNAPSHOT-shaded.jar
    -i my_input_problem.p
    -o my_output_dir
    -constants rigid
    -domains constant,cumulative
    -consequences global
    -systems T,S4,S5
```

Embed problem `my_input_problem.p` which contains a logic specification and select a particular encoding of the same semantics by applying transformation parameters:

```
    java -jar embed-1.0-SNAPSHOT-shaded.jar
    -i my_input_problem.p
    -o my_output_problem.p
    -t syntactic_modality_axiomatization,syntactic_cumulative_quantification
```

References

[1] Henk Barendregt, Wil Dekkers, and Richard Statman. *Lambda Calculus with Types*. Perspectives in Logic. Cambridge University Press, 2013.

[2] Christoph Benzmüller. Automating access control logic in simple type theory with LEO-II. In Dimitris Gritzalis and Javier López, editors, *Emerging Challenges for Security, Privacy and Trust, 24th IFIP TC 11 International Information Security Conference, SEC 2009, Pafos, Cyprus, May 18-20, 2009. Proceedings*, volume 297 of *IFIP*, pages 387–398. Springer, 2009. Url (preprint): http://christoph-benzmueller.de/papers/C27.pdf.

[3] Christoph Benzmüller, Chad Brown, and Michael Kohlhase. Higher-order semantics and extensionality. *Journal of Symbolic Logic*, 69(4):1027–1088, 2004.

[4] Christoph Benzmüller, Ali Farjami, and Xavier Parent. Aqvist's dyadic deontic logic e in hol, 2018. Url (preprint): http://orbilu.uni.lu/handle/10993/37014.

[5] 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.

[6] Christoph Benzmüller and Lawrence Paulson. Multimodal and intuitionistic logics in simple type theory. *The Logic Journal of the IGPL*, 18(6):881–892, 2010. Url (preprint): http://christoph-benzmueller.de/papers/J21.pdf.

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

[8] Christoph Benzmüller, Florian Rabe, and Geoff Sutcliffe. THF0 – the core of the TPTP language for classical higher-order logic. In Alessandro Armando, Peter Baumgartner, and Gilles Dowek, editors, *Automated Reasoning, 4th International Joint Conference, IJCAR 2008, Sydney, Australia, August 12-15, 2008, Proceedings*, volume 5195 of *LNCS*, pages 491–506. Springer, 2008. Url (preprint): http://christoph-benzmueller.de/papers/C25.pdf.

[9] 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.

[10] Christoph Benzmüller and Dana Scott. Automating free logic in Isabelle/HOL. In G.-M. Greuel, T. Koch, P. Paule, and A. Sommese, editors, *Mathematical Software –*

ICMS 2016, 5th International Congress, Proceedings, volume 9725 of *LNCS*, Berlin, Germany, 2016. Springer. To appear.

[11] Christoph Benzmüller, Leon Weber, and Bruno Woltzenlogel Paleo. Computer-assisted analysis of the anderson-hájek controversy. *Logica Universalis*, 11(1):139–151, 2017.

[12] Christoph Benzmüller and Bruno 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. (Invited paper, mildly reviewed).

[13] Christoph Benzmüller and Bruno 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. (Invited paper, mildly reviewed).

[14] Christoph Benzmüller and Bruno Woltzenlogel Paleo. The inconsistency in Gödel's ontological argument: A success story for AI in metaphysics. In Subbarao Kambhampati, editor, *IJCAI 2016*, volume 1-3, pages 936–942. AAAI Press, 2016. (Acceptance rate $\leq$ 25%).

[15] Jasmin Christian Blanchette and Tobias Nipkow. Nitpick: A ounterexample generator for higher-order logic based on a relational model finder. In M. Kaufmann and L. Paulson, editors, *Interactive Theorem Proving (ITP 2010)*, volume 6172 of *LNCS*, pages 131–146. Springer, 2010.

[16] Chad E. Brown. Satallax: An automatic higher-order prover. In Bernhard Gramlich, Dalef Miller, and Uli Sattler, editors, *Automated Reasoning*, pages 111–117, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[17] Chad E. Brown. *Satallax: An Automatic Higher-Order Prover*, pages 111–117. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[18] Walter Carnielli, Marcelo Coniglio, Dov M. Gabbay, Paula Gouveia, and Cristina Sernadas. *Analysis and synthesis of logics*, volume 35 of *Applied Logic Series*. Springer, Dordrecht, 2008.

[19] A. Church. A formulation of the simple theory of types. In *Journal of Symbolic Logic*, page 5:56–68, 1940.

[20] Chad E. Brown, Thibault Gauthier, Cezary Kaliszyk, Geoff Sutcliffe, and Josef Urban. Grunge: A grand unified atp challenge, 03 2019.

[21] Ali Farjami, Paul Meder, Xavier Parent, and Christoph Benzmüller. I/o logic in hol, 2018. Url (preprint): http://orbilu.uni.lu/handle/10993/37013.

[22] Melvin Fitting. Handbook of logic in artificial intelligence and logic programming (vol. 1). chapter Basic Modal Logic, pages 368–448. Oxford University Press, Inc., New York, NY, USA, 1993.

[23] Melvin Fitting and Richard L. Mendelsohn. *First-order Modal Logic*. Kluwer Academic Publishers, Norwell, MA, USA, 1999.

[24] Gottlob Frege. *Grundgesetze der Arithmetik*, volume 1. H. Pohle, 1893.

[25] James Garson. Modal logic. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, fall 2018 edition, 2018.

[26] Tobias Gleißner. Converting Higher-Order Modal Logic Problems into Classical Higher-Order Logic. Bsc. thesis, Freie Universität Berlin, Institute of Computer Science, Berlin, Germany, 2016.

[27] Tobias Gleißner and Alexander Steen. The met: The art of flexible reasoning with modalities. In Christoph Benzmüller, Francesco Ricca, Xavier Parent, and Dumitru Roman, editors, *Rules and Reasoning*, pages 274–284, Cham, 2018. Springer International Publishing.

[28] Tobias Gleißner, Alexander Steen, and Christoph Benzmüller. Theorem provers for every normal modal logic. In Thomas Eiter and David Sands, editors, *LPAR-21. 21st International Conference on Logic for Programming, Artificial Intelligence and Reasoning*, volume 46 of *EPiC Series in Computing*, pages 14–30, Maun, Botswana, 2017. EasyChair.

[29] R. Goldblatt. *Quantifiers, Propositions and Identity: Admissible Semantics for Quantified Modal and Substructural Logics*. Lecture Notes in Logic. Cambridge University Press, 2011.

[30] Thomas Hales, Mark Adams, Gertrud Bauer, Dat Tat Dang, John Harrison, Truong Le Hoang, Cezary Kaliszyk, Victor Magron, Sean McLaughlin, Thang Tat Nguyen, Truong Quang Nguyen, Tobias Nipkow, Steven Obua, Joseph Pleso, Jason Rute, Alexey Solovyev, An Hoai Thi Ta, Trung Nam Tran, Diep Thi Trieu, Josef Urban, Ky Khac Vu, and Roland Zumkeller. A formal proof of the kepler conjecture, 2015.

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

[32] Ullrich Hustadt and Renate A. Schmidt. Mspass: Modal reasoning by translation and first-order resolution. In Roy Dyckhoff, editor, *Automated Reasoning with Analytic Tableaux and Related Methods*, pages 67–71, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[33] Marijn J. H. Heule and Oliver Kullmann. The science of brute force. *Communications of the ACM*, 60:70–79, 07 2017.

[34] Cezary Kaliszyk, Geoff Sutcliffe, and Florian Rabe. Th1: The tptp typed higher-order form with rank-1 polymorphism. In *PAAR@IJCAR*, 2016.

[35] Cezary Kaliszyk and Josef Urban. Hol(y)hammer: Online atp service for hol light. *Mathematics in Computer Science*, 9(1):5–22, Mar 2015.

[36] Saul A. Kripke. Semantical analysis of modal logic i normal modal propositional calculi. *Mathematical Logic Quarterly*, 9(5-6):67–96, 1963.

[37] David K. Lewis. Counterpart theory and quantified modal logic. *The Journal of Philosophy*, 65(5):113–126, 1968.

[38] William McCune. Otter 3.3 reference manual, 11 2003.

[39] Reinhard Muskens. 10 higher order modal logic. In Johan Van Benthem Patrick Blackburn and Frank Wolter, editors, *Handbook of Modal Logic*, volume 3 of *Studies in Logic and Practical Reasoning*, pages 621 – 653. Elsevier, 2007.

[40] Open logic project contributors. Normal modal logic. Available at `http://builds.openlogicproject.org/content/normal-modal-logic/normal-modal-logic.pdf`, accessed 15-August-2019.

[41] Jens Otten. Mleancop: A connection prover for first-order modal logic. In Stéphane Demri, Deepak Kapur, and Christoph Weidenbach, editors, *Automated Reasoning*, pages 269–276, Cham, 2014. Springer International Publishing.

[42] T. Raths and J. Otten. The QMLTP Problem Library for First-Order Modal Logics. In B. Gramlich, D. Miller, and U. Sattler, editors, *IJCAR 2012*, volume 7364 of *LNCS*, pages 454–461. Springer, 2012.

[43] Renate Schmidt. Advances in modal logic. Available at `http://www.cs.man.ac.uk/~schmidt/tools/`, accessed 06-February-2019].

[44] Henrik Sahlqvist. Completeness and correspondence in the first and second order semantics for modal logic. *Studies in Logic and the Foundations of Mathematics*, 82:110–143, 1975.

[45] Moses Schönfinkel. Über die bausteine der mathematischen logik. *Mathematische annalen*, 92(3-4):305–316, 1924.

[46] J.H. Sobel. Appx. b: Notes in dana scott's hand. In *Logic and Theism: Arguments for and Against Beliefs in God*, page 145–146. Cambridge University Press, 2004.

[47] Alexander Steen and Christoph Benzmüller. Sweet SIXTEEN: Automation via embedding into classical higher-order logic. *Logic and Logical Philosophy*, 2016. To appear.

[48] Alexander Steen and Christoph Benzmüller. System demonstration: The higher-order prover Leo-III. In Christoph Benzmüller and Jens Otten, editors, *ARQNL 2018. Automated Reasoning in Quantified Non-Classical Logics*, volume 2095, pages 79–85. CEUR Workshop Proceedings, http://ceur-ws.org, 2018. `http://ceur-ws.org/Vol-2095/`.

[49] Aaron Stump, Geoff Sutcliffe, and Cesare Tinelli. Starexec: A cross-community infrastructure for logic solving. In Stéphane Demri, Deepak Kapur, and Christoph Weidenbach, editors, *Automated Reasoning*, pages 367–373, Cham, 2014. Springer International Publishing.

[50] G. Sutcliffe. The CADE ATP System Competition - CASC. *AI Magazine*, 37(2):99–101, 2016.

[51] G. Sutcliffe. The TPTP Problem Library and Associated Infrastructure. From CNF to TH0, TPTP v6.4.0. *Journal of Automated Reasoning*, 59(4):483–502, 2017.

[52] Geoff Sutcliffe. The cade atp system competition. Available at `http://www.tptp.org/CASC/`, accessed 20-August-2019].

[53] L.C. Paulson T. Nipkow and M. Wenzel. Isabelle/hol: A proof assistant for higher-order logic. In *Number 2283 in LNCS*, 2002.

[54] J. Van Heijenoort. *From Frege to Gödel: A Source Book in Mathematical Logic, 1879-1931*. Source books in the history of the sciences. Harvard University Press, 1967.

[55] Wikipedia contributors. Formal verification. Available at `https://en.wikipedia.org/wiki/Formal_verification`, accessed 06-February-2019.

[56] M. Wisniewski and A. Steen. Embedding of Quantified Higher-Order Nominal Modal Logic into Classical Higher-Order Logic. In C. Benzmüller and J. Otten, editors, *Automated Reasoning in Quantified Non-Classical Logics (ARQNL), Proceedings*, volume 33 of *EPiC*, pages 59–64. EasyChair, 2014.

[57] Max Wisniewski, Alexander Steen, and Christoph Benzmüller. Tptp and beyond: Representation of quantified non-classical logics. In Christoph Benzmüller and Jens Otten, editors, *ARQNL 2016. Automated Reasoning in Quantified Non-Classical Logics*, 2016.