Modal Satisfiability via SMT Solving

Carlos Areces, Pascal Fontaine, Stephan Merz

Modal Satisfiability via SMT Solving*

Carlos Areces[1], Pascal Fontaine[2,3], and Stephan Merz[2,3]

[1] Universidad Nacional de Córdoba & CONICET, Córdoba, Argentina
[2] Université de Lorraine, LORIA, UMR 7503, Vandœvre-lès-Nancy, France
[3] INRIA, Villers-lès-Nancy, France

Abstract. Modal logics extend classical propositional logic, and they are robustly decidable. Whereas most existing decision procedures for modal logics are based on tableau constructions, we propose a framework for obtaining decision procedures by adding instantiation rules to standard SAT and SMT solvers. Soundness, completeness, and termination of the procedures can be proved in a uniform and elementary way for the basic modal logic and some extensions.

1 Introduction

Classical languages like first-order and second-order logic have been investigated in detail, and their model theory is well developed. Computationally though, they are lacking as their satisfiability problem is undecidable [9,25], and even their model checking problem is already PSPACE-complete [8]. This has motivated a search for computationally well-behaved fragments. For instance, early in the twentieth century, Löwenheim already gave a decision procedure for the satisfiability of first-order sentences with only unary predicates [17]. Many familiar fragments of first-order logic are defined by means of restrictions of the quantifier prefix of formulas in prenex normal forms, and their (un)decidability has been carefully charted [7]. Finite-variable (and in particular two-variable) fragments of first-order logic are yet another kind of fragments whose computational properties have been studied extensively, with decidability results going back to the early 1960s [23,14,15,20]. But even though many of these fragments have good computational properties, their meta-logical properties are often poor, and, in particular, they usually lack a good model theory that explains their computational properties.

Research efforts have been devoted to identify fragments of first-order or second-order logic that manage to combine good computational behavior with good meta-logical properties. One such effort takes (propositional) modal logic as its starting point [4,5]. Although modal logics are syntactically presented as an extension of propositional logic, there exist well-known translations through which modal languages may semantically be viewed as fragments of first-order languages. Modal fragments are

* This work was partly supported by grants ANPCyT-PICT-2008-306, ANPCyT-PICT-2010-688, ANPCyT-PICT-2012-712, the FP7-PEOPLE-2011-IRSES Project "Mobility between Europe and Argentina applying Logics to Systems" (MEALS), the project ANR-13-IS02-0001 of the Agence Nationale de la Recherche, the STIC AmSud MISMT, and the Laboratoire International Associé "INFINIS".

computationally very well-behaved; their satisfiability and model checking problems are of reasonably low complexity, and they are so in a robust way [26,13]. The good computational behavior of modal fragments has been explained in terms of the tree model property, and generalizations thereof.

Broadly speaking, there are three general strategies for modal theorem proving: *(i)* develop purpose-built calculi and tools; *(ii)* translate modal problems into automata-theoretic problems, and use automata-theoretic methods to obtain answers; *(iii)* translate modal problems into first-order problems, and use general first-order tools. The advantage of indirect methods such as *(ii)* and *(iii)* is that they allow us to reuse well-developed and well-supported tools instead of having to develop new ones from scratch. In this paper we focus on the third option: translation-based theorem proving for modal logic, where modal formulas are translated into first-order formulas to be fed to first-order theorem provers. In particular, we will investigate the use of Satisfiability Modulo Theories (SMT) techniques [3,18] for reasoning in restricted first-order theories. We provide rules that constrain the instantiations of quantifiers in the translated formulas, and we show that these rules are sound, complete and terminating.

Outline. Section 2 introduces basic modal logic and explains the overall architecture of the SMT-based decision procedure. The precise rules are indicated in section 3, together with the proofs of soundness, completeness, and termination. Extensions of the procedure to global modalities and hybrid logic appear in section 4, and section 5 concludes the paper and discusses related work.

2 Background

2.1 Basic Modal Logic

The basic modal logic BML can be seen as an extension of propositional logic. Let $\mathcal{P}$ be a set of propositional symbols, the syntax of BML is defined as

$$\varphi \doteq p \mid \neg p \mid \varphi \wedge \varphi' \mid \varphi \vee \varphi \mid \Box\varphi \mid \Diamond\varphi,$$

where $p \in \mathcal{P}$. Observe that we assume formulas to be in negation normal form where negation is only applied to atomic propositions. Semantically, formulas of BML are interpreted over relational structures. Let $\mathcal{M} = \langle M, \cdot^{\mathcal{M}} \rangle$ be such that M is a non-empty set called the *domain* and $\cdot^{\mathcal{M}}$ is an *interpretation function* that assigns to each $p \in \mathcal{P}$ a subset $p^{\mathcal{M}}$ of M and introduces a relation $R^{\mathcal{M}} \subseteq M \times M$ ($R^{\mathcal{M}}$ is usually called the *accessibility relation* of $\mathcal{M}$). For a relational structure $\mathcal{M}$, we will often write $|\mathcal{M}|$ for the domain of $\mathcal{M}$, and if $w \in |\mathcal{M}|$ we will say that $\mathcal{M}, w$ is a pointed model. For a pointed model $\mathcal{M}, w$, the satisfaction relation for formulas in BML is defined by

$$\begin{array}{rll}
\mathcal{M}, w \models p & \text{iff} & w \in p^{\mathcal{M}} \\
\mathcal{M}, w \models \neg p & \text{iff} & w \notin p^{\mathcal{M}} \\
\mathcal{M}, w \models \varphi \wedge \varphi' & \text{iff} & \mathcal{M}, w \models \varphi \text{ and } \mathcal{M}, w \models \varphi' \\
\mathcal{M}, w \models \varphi \vee \varphi' & \text{iff} & \mathcal{M}, w \models \varphi \text{ or } \mathcal{M}, w \models \varphi' \\
\mathcal{M}, w \models \Box\varphi & \text{iff} & \text{for all } (w,v) \in R^{\mathcal{M}} \text{ we have that } \mathcal{M}, v \models \varphi \\
\mathcal{M}, w \models \Diamond\varphi & \text{iff} & \text{for some } (w,v) \in R^{\mathcal{M}} \text{ we have that } \mathcal{M}, v \models \varphi.
\end{array}$$

We say that φ is *satisfiable* if there is a pointed model $\mathcal{M},w$ such that $\mathcal{M},w \models \varphi$, otherwise φ is unsatisfiable. We define $\mathsf{Mod}(\varphi)$ as the set of pointed models of φ, formally, $\mathsf{Mod}(\varphi) = \{\mathcal{M},w \mid \mathcal{M},w \models \varphi\}$ (we will use $\mathsf{Mod}(\varphi)$ for the set of models of φ also when φ is a propositional formula). For a set Σ of formulas, we let $\mathsf{Mod}(\Sigma) = \bigcap_{\varphi\in\Sigma} \mathsf{Mod}(\varphi)$. Finally, let $\Sigma \cup \{\varphi\}$ be a set of formulas, we say that φ is a consequence of Σ and we write $\Sigma \models \varphi$ if $\mathsf{Mod}(\Sigma) \subseteq \mathsf{Mod}(\varphi)$.

The definition above makes it clear that the semantics of basic modal logic is purely first-order. Actually, through *translation*, modal languages may be viewed as fragments of first-order languages. Our starting point is the relational translation ST, which translates modal formulas by transcribing their truth definitions as first-order formulas. Let φ be a modal formula and x a first-order variable; then $\mathsf{ST}_x(\varphi)$ is defined as follows:

$$\mathsf{ST}_x(p) \doteq P(x) \qquad \mathsf{ST}_x(\neg p) \doteq \neg P(x) \tag{1}$$

$$\mathsf{ST}_x(\varphi \wedge \varphi') \doteq \mathsf{ST}_x(\varphi) \wedge \mathsf{ST}_x(\varphi') \qquad \mathsf{ST}_x(\varphi \vee \varphi') \doteq \mathsf{ST}_x(\varphi) \vee \mathsf{ST}_x(\varphi') \tag{2}$$

$$\mathsf{ST}_x(\Box\varphi) \doteq \forall y : \neg R(x,y) \vee \mathsf{ST}_y(\varphi) \qquad \mathsf{ST}_x(\Diamond\varphi) \doteq \exists y : R(x,y) \wedge \mathsf{ST}_y(\varphi). \tag{3}$$

In (1), P is a unary predicate symbol corresponding to the proposition letter p; in (3), the variable y is fresh. Observe how (3) reflects the truth definition of the modal operators. Also observe that $\mathsf{ST}_x(\varphi)$ is a first-order formula in negation normal form whose only free variable is x, for any BML formula φ. ST is extended to sets of formulas in the obvious way, i.e., $\mathsf{ST}_x(\Sigma) = \{\mathsf{ST}_x(\varphi) : \varphi \in \Sigma\}$.

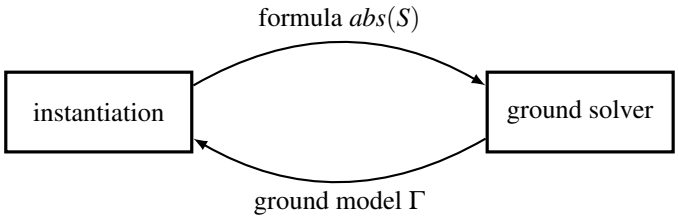
Proposition 1. *Let $\mathcal{M},w$ be a pointed model, and φ a formula of BML then*

$$\mathcal{M},w \models \varphi \quad \textit{iff} \quad \mathcal{M}, g[x \mapsto w] \models \mathsf{ST}_x(\varphi),$$

where, on the right-hand side, $\mathcal{M}$ is viewed as a first-order interpretation, g is an arbitrary assignment for $\mathcal{M}$, and $g[x \mapsto w]$ coincides with g but assigns w to x.

2.2 SMT Solving for Modal Satisfiability: Overall Setup

Starting from the relational translation of modal logic into first-order logic, we propose in this paper an SMT-based procedure for deciding the satisfiability of BML formulas. It consists of two cooperating modules, as illustrated in the schema below.



The procedure maintains a finite set S of first-order formulas. Initially, S is obtained by relational translation from the set of BML formulas whose satisfiability we wish to decide. New formulas can be added through instantiation, provided that the resulting set of formulas is equisatisfiable with the original one.

The *ground solver* is given a ground abstraction of the set S, denoted in the following by $abs(S)$, and it decides if $abs(S)$ is satisfiable. The solver is assumed to be sound, complete, and terminating for ground formulas; it includes a SAT solver and possibly other decision procedures. For example, the extension to hybrid logic of section 4.2 requires a decision procedure for quantifier-free formulas over the theory of uninterpreted function and predicate symbols with equality. The abstraction $abs(S)$ for BML is obtained by (consistently) mapping formulas φ in S that are either atomic or contain an outermost quantifier to fresh propositional symbols that we denote by $\boxed{\varphi}$, while preserving the Boolean structure of formulas in S. For example, consider the set

$$S = \{(\forall x : \neg R(c,x) \vee P(x)),\ R(c,d) \wedge \neg P(d)\}.$$

Its abstraction will be denoted as

$$abs(S) = \{\boxed{\forall x : \neg R(c,x) \vee P(x)},\ \boxed{R(c,d)} \wedge \neg \boxed{P(d)}\}.$$

Conversely, given a set Γ of propositional formulas whose atoms are all of the form $\boxed{\varphi}$, we define $conc(\Gamma)$ to be the set of first-order formula obtained by "erasing the boxes". In particular, $conc(abs(S)) = S$.

If the ground solver finds $abs(S)$ to be unsatisfiable, then the procedure declares the original set of formulas unsatisfiable. Otherwise, it computes a set of literals Γ built from the atomic propositions in $abs(S)$ that corresponds to a model of $abs(S)$. Although Γ is a propositional model of $abs(S)$, a first-order model for $conc(\Gamma)$, or indeed for S, may not exist. For example, the set S above is unsatisfiable, but $abs(S)$ has the propositional model

$$\Gamma = \{\boxed{\forall x : \neg R(c,x) \vee P(x)},\ \boxed{R(c,d)}, \neg \boxed{P(d)}\}.$$

In order to rule out models of $abs(S)$ that do not correspond to first-order models of S, the decision procedure also contains an *instantiation module* that computes refinements of $abs(S)$. More precisely, given a model Γ of $abs(S)$, the instantiation module may generate relevant instances of the quantified formulas that are abstracted in $abs(S)$. For the above example, the instantiation module should produce the formula

$$(\forall x : \neg R(c,x) \vee P(x)) \Rightarrow (\neg R(c,d) \vee P(d))$$

that will be added to S, yielding set S'. Note that Γ is no longer a model of $abs(S')$, and that in fact $abs(S')$ is unsatisfiable. We must ensure that only finitely many instances are generated over time, so that the feedback loop eventually terminates and the procedure outputs a verdict.

3 Decision Procedure for Basic Modal Logic

We now define instantiation rules for basic modal logic BML. We show that these rules are sound and complete. Moreover, only finitely many instantiations are created for each quantified formula, hence the procedure terminates.

3.1 Instantiation Rules

Recall from section 2.1 that formulas arising from the relational translation of basic modal logic are built from unary predicate symbols $P(\_)$ that correspond to the proposition symbols in $\mathcal{P}$ and a single binary predicate symbol $R(\_,\_)$. Formulas are in negation normal form and contain exactly one free variable, representing the current point of evaluation. Occurrences of quantifiers are restricted to the forms

$$\forall y : \neg R(x,y) \vee \varphi(y) \qquad \text{and} \qquad \exists y : R(x,y) \wedge \varphi(y)$$

where x is the variable designating the point of evaluation of the quantified formula, and y is the only free variable in $\varphi(y)$.

The set of formulas given as input to the decision procedure is assumed to consist of formulas of this form. We replace the unique free variable by a Skolem constant, obtaining a closed set S^0 of formulas. The decision procedure will maintain a set of formulas in this form, extended by formulas $(Qx : \psi(x)) \Rightarrow \chi$ where Q is a quantifier and $\psi(x)$ and χ are formulas in negation normal form; χ has no free variables and $\psi(x)$ contains exactly the free variable x.

A *configuration* of the decision procedure is a triple $\langle S, \Theta_\exists, \Theta_\forall \rangle$ where

- S is a set of closed formulas as described above and
- the sets $\Theta_\exists$ and $\Theta_\forall$ contain information about instances that have already been produced and need not be created anew.

The initial configuration is $\langle S^0, \emptyset, \emptyset \rangle$. Given a configuration $\langle S, \Theta_\exists, \Theta_\forall \rangle$, the decision procedure invokes the ground solver on the set $abs(S)$. If $abs(S)$ is unsatisfiable, the procedure terminates, declaring S^0 unsatisfiable. Otherwise, the ground solver produces a set of literals Γ that represents a model of $abs(S)$, and the decision procedure computes a successor configuration by applying one of the following instantiation rules $(\exists)$ or $(\forall)$ and continues. If no instantiation rule is applicable, the procedure terminates, declaring S^0 satisfiable.

Rule $(\exists)$. This rule instantiates existentially quantified formulas in S by fresh constants:

$\langle S, \Theta_\exists, \Theta_\forall \rangle \xrightarrow{\Gamma} \langle S', \Theta'_\exists, \Theta_\forall \rangle$ if there exists $\varepsilon \doteq \boxed{\exists y : R(c,y) \wedge \varphi(y)}$ s.t.

- $\varepsilon \in \Gamma \setminus \Theta_\exists$ is an atom corresponding to an existentially quantified formula that appears in Γ but for which no instance has yet been created,
- d is a fresh constant,
- $S' = S \cup \{conc(\varepsilon) \Rightarrow (R(c,d) \wedge \varphi(d))\}$,
- $\Theta'_\exists = \Theta_\exists \cup \{\varepsilon\}$.

Rule $(\forall)$. This rule instantiates universally quantified formulas in S for constants such that the guard of the quantified formula appears in Γ:

$\langle S, \Theta_\exists, \Theta_\forall \rangle \xrightarrow{\Gamma} \langle S', \Theta_\exists, \Theta'_\forall \rangle$ if there exist $\varepsilon \doteq \boxed{\forall y : \neg R(c,y) \vee \varphi(y)}$ and d s.t.

- $\varepsilon \in \Gamma$, $\boxed{R(c,d)} \in \Gamma$, $(\varepsilon, d) \notin \Theta_\forall$. In words, ε is an atom that corresponds to a universally quantified formula that appears in Γ, and d is a constant for which the guard of ε is asserted in Γ but for which ε has not yet been instantiated,
- $S' = S \cup \{conc(\varepsilon) \Rightarrow (\neg R(c,d) \vee \varphi(d))\}$,
- $\Theta'_\forall = \Theta_\forall \cup \{(\varepsilon, d)\}$.

The rules are natural, and they resemble the rules in a tableaux algorithm for the basic modal language, but implemented in the SMT setup. In particular, rule $(\exists)$ uses fresh constants to denote unique witnesses for existential quantifiers. It can be understood as on-the-fly Skolemization of an outermost existential quantifier. Universal quantifiers are instantiated only for successors (via the accessibility relation) of the only constant that appears in the guard of the quantifier. These instantiations are guided by the propositional model Γ computed by the ground solver.

3.2 Soundness and Completeness

The soundness of the rules $(\exists)$ and $(\forall)$ is a consequence of the following two lemmas, whose proof is straightforward.

Lemma 2. *Assume that* $\langle S, \Theta_\exists, \Theta_\forall \rangle \xrightarrow{\Gamma} \langle S', \Theta'_\exists, \Theta'_\forall \rangle$ *according to rules* $(\exists)$ *or* $(\forall)$. *Then* S *and* S' *are equisatisfiable sets of first-order formulas.*

Lemma 3. *If* $\mathcal{M}$ *is a first-order model of* S, *then* $abs(S)$ *has a ground model* Γ.

Proof. Define Γ to be the set of literals built from the atomic formulas in $abs(S)$ such that $\boxed{\psi} \in \Gamma$ if $\mathcal{M} \models \psi$ and $\neg\boxed{\psi} \in \Gamma$ if $\mathcal{M} \not\models \psi$. A straightforward inductive proof shows that $\Gamma \models abs(S)$. □

Theorem 4 (Soundness). *Assume that the procedure terminates with verdict "unsatisfiable". Then the initial set* S^0 *of formulas is unsatisfiable.*

Proof. The verdict "unsatisfiable" is based on a sequence of configurations

$$\langle S^0, \emptyset, \emptyset \rangle \xrightarrow{\Gamma^0} \langle S^1, \Theta^1_\exists, \Theta^1_\forall \rangle \xrightarrow{\Gamma^1} \cdots \xrightarrow{\Gamma^{n-1}} \langle S^n, \Theta^n_\exists, \Theta^n_\forall \rangle$$

such that the ground solver finds $abs(S^n)$ to be unsatisfiable. By Lemma 3, it follows that S^n is unsatisfiable, and so is S^0, by iterating Lemma 2. □

The completeness proof relies on the construction of a first-order model of the original set S^0 of formulas from a propositional model of a saturated set S^n in a configuration where no rules are applicable anymore.

Theorem 5 (Completeness). *Assume that the procedure terminates with verdict "satisfiable". Then the initial set* S^0 *of formulas is satisfiable.*

Proof. The verdict "satisfiable" is based on a sequence of configurations

$$\langle S^0, \emptyset, \emptyset \rangle \xrightarrow{\Gamma^0} \langle S^1, \Theta^1_\exists, \Theta^1_\forall \rangle \xrightarrow{\Gamma^1} \cdots \xrightarrow{\Gamma^{n-1}} \langle S^n, \Theta^n_\exists, \Theta^n_\forall \rangle$$

such that the ground solver finds $abs(S^n)$ to be satisfiable, and no transition according to the rules $(\exists)$ or $(\forall)$ is possible. The ground solver produces a set Γ of literals that corresponds to a propositional model of $abs(S^n)$; more precisely, the propositional interpretation Γ^* that satisfies the atomic formulas $\boxed{\psi}$ iff $\boxed{\psi} \in \Gamma$ is a model of $abs(S^n)$.

Observe that the set S^n is a superset of S^0 obtained by adding formulas of the form

$$(\exists y : R(c,y) \wedge \varphi(y)) \Rightarrow \ldots \qquad \text{and} \qquad (\forall y : \neg R(c,y) \vee \varphi(y)) \Rightarrow \ldots$$

by applications of rules $(\exists)$ and $(\forall)$. We define a first-order interpretation $\mathcal{M}$ as follows. The universe $|\mathcal{M}|$ consists of the constants that appear in S^n: observe that this set is non-empty since S^0 contains precisely one constant and $S^0 \subseteq S^n$. For the predicate symbols $P(\_)$ that appear in S^n and any $a \in |\mathcal{M}|$, we define $a \in P^{\mathcal{M}}$ iff $\boxed{P(a)} \in \Gamma$. Similarly, for the relation symbol, we let $(a,b) \in R^{\mathcal{M}}$ iff $\boxed{R(a,b)} \in \Gamma$.

Step 1. We show that for every ground instance of every subformula ψ of a formula in S^0 for constants that appear in S^n, if $\Gamma^* \models abs(\psi)$ then $\mathcal{M} \models \psi$. The proof is by induction on ψ.

- For $\psi \doteq P(a)$ or $\psi \doteq R(a,b)$, if $\Gamma^* \models abs(\psi)$ then $abs(\psi) \in \Gamma$ and therefore $\mathcal{M} \models \psi$ by definition of $\mathcal{M}$. Similarly, if $\Gamma^* \models \neg abs(\psi)$ then $abs(\psi) \notin \Gamma$ and therefore $\mathcal{M} \models \neg\psi$, again by definition of $\mathcal{M}$.
- For conjunctions and disjunctions, the proof of the inductive step is immediate.
- Assume that $\psi \doteq \exists y : R(c,y) \wedge \varphi(y)$ is a ground instance of a subformula in S^0 and that $\Gamma^* \models abs(\psi)$, i.e. $\boxed{\psi} \in \Gamma$. Since rule $(\exists)$ cannot be applied, we must have $\boxed{\psi} \in \Theta_\exists$, and therefore S^n must contain

  $$\psi \Rightarrow (R(c,d) \wedge \varphi(d))$$

  for some constant d, where the right-hand side of the implication is a ground instance of a subformula in S^0. Moreover, $abs(S^n)$ contains

  $$\boxed{\psi} \Rightarrow abs(R(c,d) \wedge \varphi(d)).$$

  Since $\Gamma^* \models \boxed{\psi}$, it follows that $\Gamma^* \models abs(R(c,d) \wedge \varphi(d))$. Now, $\mathcal{M} \models R(c,d) \wedge \varphi(d)$ follows by induction hypothesis, and this proves $\mathcal{M} \models \psi$.
- Assume now that $\psi \doteq \forall y : \neg R(c,y) \vee \varphi(y)$ and that $\Gamma^* \models abs(\psi)$, i.e. $\boxed{\psi} \in \Gamma$. Moreover, assume that $\mathcal{M} \models R(c,d)$ for a constant $d \in |\mathcal{M}|$: we must show that $\mathcal{M} \models \varphi(d)$. By the definition of $\mathcal{M}$ and the assumption that $\mathcal{M} \models R(c,d)$ it follows

that $\boxed{R(c,d)} \in \Gamma$. Since rule $(\forall)$ cannot be applied, we must have $(\boxed{\psi}, d) \in \Theta_\forall$, and S^n, resp. $abs(S^n)$, contain the formulas

$$\psi \Rightarrow (\neg R(c,d) \vee \varphi(d)) \qquad \text{resp.} \qquad \boxed{\psi} \Rightarrow \big(\neg \boxed{R(c,d)} \vee abs(\varphi(d))\big).$$

where the right-hand side of the implication on the left is a ground instance of a subformula of S^0. Because $\Gamma^* \models abs(S^n)$, we can conclude that $\Gamma^* \models abs(\varphi(d))$, and therefore $\mathcal{M} \models \varphi(d)$ by induction hypothesis, and this suffices.

Step 2. Now suppose that formula φ appears in the original set S^0. Then φ is ground and $\Gamma^* \models abs(\varphi)$ because Γ^* is a model of $abs(S^n) \supseteq abs(S_0)$. By step 1, it follows that $\mathcal{M} \models \varphi$. Thus, $\mathcal{M}$ is a model of S^0, and this concludes the proof. □

Remark. Notice that the restriction to ground instances of (sub-)formulas of S^0 in the above proof is necessary: the model $\mathcal{M}$ need not satisfy all formulas in S^n. For example, consider a set S^0 containing the formula

$$\underbrace{(\exists x : R(a,x) \wedge P(x))}_{\varepsilon_1} \vee \underbrace{(\exists y : R(a,y) \wedge P(y) \wedge Q(y))}_{\varepsilon_2}$$

resulting from the translation of the modal formula $(\Diamond p) \vee \Diamond(p \wedge q)$. The saturation of S^0 by application of the instantiation rules may result in a set S^n containg the two implications

$$\varepsilon_1 \Rightarrow R(a,b) \wedge P(b) \quad \text{and} \quad \varepsilon_2 \Rightarrow R(a,c) \wedge P(c) \wedge Q(c)$$

for two constants b and c. A possible propositional model Γ of $abs(S^n)$ contains the literals

$$\neg\boxed{\varepsilon_1},\ \neg\boxed{R(a,b)},\ \neg\boxed{P(b)},\ \boxed{\varepsilon_2},\ \boxed{R(a,c)},\ \boxed{P(c)},\ \boxed{Q(c)}.$$

The corresponding first-order interpretation satisfies $R(a,c) \wedge P(c)$, hence it satisfies ε_1, but it does not satisfy $R(a,b)$ or $P(b)$. Therefore it is not a model of S^n.

Observe, however, that the formulas added by applications of rule $(\forall)$ are first-order valid, and in particular true in the interpretation $\mathcal{M}$.

3.3 Termination

Finally, we show that the procedure must terminate because only finitely many constants can be introduced during any run of the procedure.

Theorem 6 (Termination). *For any finite set S^0, there cannot be an infinite transition sequence*

$$\langle S^0, \Theta^0_\exists, \Theta^0_\forall \rangle \xrightarrow{\Gamma^0} \langle S^1, \Theta^1_\exists, \Theta^1_\forall \rangle \xrightarrow{\Gamma^1} \ldots$$

Proof. The key is to observe that only finitely many constants can be introduced in sets S^i by applications of rule $(\exists)$, and that every constant can only give rise to finitely many applications of rule $(\forall)$. We associate a depth ∂_c with every constant c that appears in sets S^i, as follows:

- S^0 contains only a single constant c whose depth ∂_c is 0.
- If $\langle S^{i+1}, \Theta_\exists^{i+1}, \Theta_\forall^{i+1} \rangle$ results from an application of rule $(\forall)$ the set of constants is unchanged.
- If $\langle S^{i+1}, \Theta_\exists^{i+1}, \Theta_\forall^{i+1} \rangle$ introduces constant d through an application of rule $(\exists)$ for formula $\exists y : R(c,y) \wedge \varphi(y)$, then $\partial_d = \partial_c + 1$.

We prove by induction that the set of constants c at depth $\partial_c = k$ is finite, for any $k \in \mathbb{N}$:

- The assertion is obvious for $k = 0$.
- Assuming there are only finitely many constants c at depth k, there can only be a finite set of formula instances $\exists y : R(c,y) \wedge \varphi(y)$ in the sets S^i for every such c since all these instances come from subformulas of the original set S^0 of formulas, of which there are only finitely many, and each of these instances can be used only once to generate a new constant by rule $(\exists)$ because its abstraction is then added to $\Theta_\exists$. Therefore the set of constants of depth $k+1$ is again finite.

Moreover, the depth of constants introduced in any set S^i is bounded by the maximal quantifier depth of any formula in S^0, since every instantiation removes a quantifier.

Hence, the set of constants that appear throughout the transition sequence is finite, and therefore the rule $(\exists)$ can be applied only finitely often. Moreover, rule $(\forall)$ can only be applied once per pair of universally quantified formula instance and constant. This proves termination. □

The proof above is a recast of the standard termination proof used in tableau calculi. We now consider some extensions of the basic modal language. Interestingly, the proof requires only small changes. By comparison, the corresponding termination proof for, say, the basic hybrid logic is much more involved.

4 Extensions of the Basic Modal Logic

In this section, we consider some extensions of the basic modal logic to which we adapt the procedure described before.

4.1 Global Modalities

The relational translation for modal operators of BML gives rise to formulas where quantifiers are guarded by accessibility conditions (see definition clauses (3) on page 3). Global modalities [12] refer to arbitrary elements of the relational structure, which need not be related to the current point. The existential global modality is usually denoted by E and A is its univesal dual. Their semantics conditions are as follows

$$\begin{array}{lll} \mathcal{M}, w \models \mathsf{A}\varphi & \text{iff} & \text{for all } v \in M \text{ we have that } \mathcal{M}, v \models \varphi \\ \mathcal{M}, w \models \mathsf{E}\varphi & \text{iff} & \text{for some } v \in M \text{ we have that } \mathcal{M}, v \models \varphi. \end{array}$$

Their relational translations introduces formulas

$$\forall y : \varphi(y) \qquad \text{and} \qquad \exists y : \varphi(y)$$

where y is again the only free variable; moreover, $\varphi(y)$ does not contain any constant.

We introduce two new rules (E) and (A) for these modalities. In these rules, $\varphi(y)$ denotes an unguarded formula that contains only y as free variable (and no constant).

Rule (E). This rule instantiates unguarded existentially quantified formulas in S by fresh constants:

$\langle S, \Theta_\exists, \Theta_\forall \rangle \xrightarrow{\Gamma} \langle S', \Theta'_\exists, \Theta_\forall \rangle$ if there exists $\varepsilon \doteq \boxed{\exists y : \varphi(y)}$ s.t.

- $\varepsilon \in \Gamma \setminus \Theta_\exists$ is an atom corresponding to an unguarded existentially quantified formula that appears in Γ but has not been handled yet,
- d is a fresh constant,
- $S' = S \cup \{conc(\varepsilon) \Rightarrow \varphi(d))\}$,
- $\Theta'_\exists = \Theta_\exists \cup \{\varepsilon\}$.

Rule (A). This rule instantiates unguarded universally quantified formulas in S for constants that have not yet been instantiated.

$\langle S, \Theta_\exists, \Theta_\forall \rangle \xrightarrow{\Gamma} \langle S', \Theta_\exists, \Theta'_\forall \rangle$ if there exist $\varepsilon \doteq \boxed{\forall y : \varphi(y)}$ and d s.t.

- $\varepsilon \in \Gamma$, d is a constant in S, $(\varepsilon, d) \notin \Theta_\forall$. In words, ε is an atom corresponding to an unguarded universally quantified formula that appears in Γ, and d is a constant for which ε has not yet been instantiated,
- $S' = S \cup \{conc(\varepsilon) \Rightarrow \varphi(d)^{\mathsf{A}}\}$,
- $\Theta'_\forall = \Theta_\forall \cup \{(\varepsilon, d)\}$.

Without additional precautions, the rule (A) may lead to the regeneration of copies of formulas for different constants that could make the procedure fail to terminate. For example, a subformula $\forall x : \exists y : R(x,y) \wedge P(y)$ that corresponds to the relational translation of the modal formula $\mathsf{A}\Diamond p$ may lead to the generation of infinitely many copies of the formula $R(c,d) \wedge P(d)$ for different constants c and d.

In order to avoid the generation of redundant copies, we adopt a blocking rule similar to the one proposed by Schmidt and Tishkovsky [22]. The instantiated formula $\varphi(d)$ generated in the above rule has been decorated in order to remember that it is an instance of an unguarded universally quantifier. This decoration is understood to be distributed across the Boolean connectives that appear in φ. As a concrete example, consider the unguarded formula

$$\varepsilon \doteq \forall y : P(y) \vee (\exists z : R(y,z) \wedge Q(z))$$

that corresponds to the modal formula $\mathsf{A}(p \vee \Diamond q)$. The application of rule (A) to this formula will introduce the implication

$$conc(\varepsilon) \Rightarrow P(y)^{\mathsf{A}} \vee (\exists z : R(y,z) \wedge Q(z))^{\mathsf{A}}.$$

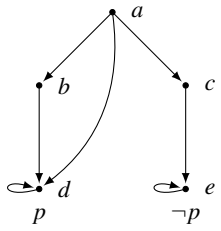


Fig. 1. Model for example formula with global modalities.

The decorated formulas are not distinguished from undecorated ones, except that we add the following variant of the rule $(\exists)$.

$\langle S, \Theta_\exists, \Theta_\forall \rangle \xrightarrow{\Gamma} \langle S', \Theta'_\exists, \Theta_\forall \rangle$ if there exists $\varepsilon \doteq \boxed{(\exists y : R(c,y) \wedge \varphi(y))^{\mathsf{A}}}$ s.t.

- $\varepsilon \in \Gamma \setminus \Theta_\exists$ is an atom corresponding to an existentially quantified formula that appears in Γ but for which no instance has yet been created,
- if S contains the formula

$$(\exists y : R(a,y) \wedge \varphi(y))^{\mathsf{A}} \Rightarrow (R(a,d) \wedge \varphi(d))$$

  for some constants a and d, then

$$S' = S \cup \{conc(\varepsilon) \Rightarrow (R(c,d) \wedge \varphi(d))\}$$

  for that constant d, otherwise S' is defined as above for a fresh constant d,
- $\Theta'_\exists = \Theta_\exists \cup \{\varepsilon\}$.

For the soundness proof, it is essential to notice that Lemma 2 carries over to the new rules. In particular, the above variant of the rule $(\exists)$ ensures equisatisfiability of sets S and S' because the same successor satisfying $\varphi(d)$ may be chosen for any two diamond formulas in the scope of a global A modality. The completeness and termination proofs of section 3 carry over to the above rules in the obvious manner.

As a concrete example, the application of our rules to the modal formula

$$\Diamond\Box p \wedge \Diamond\Box\neg p \wedge \mathsf{A}(\Diamond p \vee \Diamond\neg p)$$

may result in the model shown in figure 1. Its domain has five elements a, b, c, d, and e, with a corresponding to the root point satisfying the original formula. The proposition p is true at d and false at e. Note that every point has either d or e as a successor, ensuring that the subformula $\mathsf{A}(\Diamond p \vee \Diamond\neg p)$ is satisfied.

4.2 Hybrid Logic

Hybrid languages [2] are modal languages that have special symbols to name individual points in models. Syntactically, these new symbols $i, j, k, \ldots$, often called nominals, are

just another sort of propositional symbols[4]. For example, if i is a nominal and p and q are ordinary atomic propositions, then

$$\Box i \wedge \Diamond q \wedge \Diamond \neg q \quad \text{and} \quad \Box p \wedge \Diamond q \wedge \Diamond \neg q$$

are both well-formed formulas; but they have quite a different meaning. Actually, as we will now explain, the second formula is satisfiable whereas the first one is not. The difference comes from the interpretation that should be attributed to nominals. Because a nominal i represents a particular element in the model it should be true at a unique state. Formally, its interpretation $i^{\mathcal{M}}$ is a singleton set. For the left-hand formula above to be true at some point w, the first conjunct requires that at most one state (the one denoted by i) can be accessible from the evaluation point w. It is then impossible to satisfy both q and $\neg q$ at that unique successor, as required by the two other conjuncts. In contrast, the first conjunct of the right-hand formula just requires that all states accessible from w satisfy p, but does not restrict their multiplicity. Hence, some successor may satisfy q and another one $\neg q$.

Once we have names for states we can introduce, for each nominal i, an operator $@_i$ that allows us to jump to the point named by i. The formula $@_i\varphi$ (read "at i, φ") moves the point of evaluation to the state named by i and evaluates φ there: Intuitively, the $@_i$ operators internalize the satisfaction relation "$\models$" into the logical language:

$$\mathcal{M}, w \models @_i\varphi \quad \text{iff} \quad \mathcal{M}, u \models \varphi \text{ where } i^{\mathcal{M}} = \{u\}.$$

For this reason, these operators are usually called satisfaction operators.

We will now extend our calculus to handle the operators of the basic hybrid logic.

4.3 SMT-Based Decision Procedure for Hybrid Logic

The relational translation for basic modal logic extends to hybrid logic through the definitions

$$\begin{aligned} \mathsf{ST}_x(i) &\doteq x = i \\ \mathsf{ST}_x(@_i\varphi) &\doteq \mathsf{ST}_i(\varphi). \end{aligned}$$

In particular, note that nominals are translated as constants of first-order logic. The relational translation still produces formulas with at most one free variable.

We now adapt our SMT-based decision procedure to hybrid logic, starting from the relational translation of the input set of hybrid logic formulas; if that translation has a (single) free variable, it is again replaced by a (Skolem) constant, otherwise the translation must contain a constant corresponding to a nominal. Because equality is now a central part of reasoning, we no longer produce propositional abstractions for use with a SAT solver, but rely on a ground SMT solver that includes a decision procedure for equality over uninterpreted predicate symbols. Accordingly, the abstraction preserves

[4] Propositional symbols and nominals are however handled quite differently while translating to first-order formulas, as we will see later.

all ground formulas of the forms $P(a)$, $R(a,b)$, and $a = n$, but quantified formulas that arise from the translation of modal operators are still abstracted, as in

$$\boxed{\forall x : \neg R(c,x) \vee (x = i \wedge \neg P(x))}$$

that corresponds to asserting the formula $\Box(i \wedge \neg p)$ of hybrid logic at world c.

The algorithm from section 3 remains essentially unchanged. However, ground models Γ are now not just propositional models of $abs(S)$, but consist of an arrangement of the finite set C of constants in S given by an equivalence relation $\equiv$ whose set of equivalence classes we will denote by $[C]$, valuations $[\![P]\!] \subseteq [C]$ and $[\![R]\!] \subseteq [C] \times [C]$ that indicate the extensions of the unary and binary predicates in Γ, as well as a set of atoms $\boxed{\varphi}$ that correspond to abstracted subformulas in S that are true in Γ. The rules $(\exists)$ and $(\forall)$ of section 3 remains basically the same, except that conditions $\psi \in \Gamma$ should be read as $\Gamma \models \psi$, and that atomic formulas are no longer abstracted, as discussed above.

4.4 Soundness, Completeness, and Termination

The proof of soundness extends immediately the one of section 3.2.

Theorem 7 (Soundness for hybrid logic). *Assume that the procedure terminates with verdict "unsatisfiable". Then the initial set S^0 of formulas is unsatisfiable.*

Proof. The analogues of lemmas 2 and 3 remain true for hybrid logic: for any transition $\langle S, \Theta_\exists, \Theta_\forall \rangle \xrightarrow{\Gamma} \langle S', \Theta'_\exists, \Theta'_\forall \rangle$, we have that S and S' are equisatisfiable. Also, any first-order model of S again gives rise to a model of $abs(S)$, hence unsatisfiability of $abs(S)$ implies unsatisfiability of S.

The soundness theorem is an immediate consequence of these two lemmas. □

Completeness. The completeness proof is also analogous to the one in section 3.2: whenever the procedure produces a ground model in a state where no instantiation rule can be applied, then the set of formulas is satisfiable.

Theorem 8 (Completeness for hybrid logic). *Assume that the procedure terminates with verdict "satisfiable". Then the initial set S^0 of formulas is satisfiable.*

Proof. Assume that the procedure terminates after a sequence of transitions

$$\langle S^0, \emptyset, \emptyset \rangle \xrightarrow{\Gamma^0} \langle S^1, \Theta^1_\exists, \Theta^1_\forall \rangle \xrightarrow{\Gamma^1} \cdots \xrightarrow{\Gamma^{n-1}} \langle S^n, \Theta^n_\exists, \Theta^n_\forall \rangle$$

such that $abs(S^n)$ is satisfied by a ground model Γ, and no transition according to $(\exists)$ or $(\forall)$ is possible. As before, S^0 is exactly the subset of S^n without the formulas added by applications of rules $(\exists)$ and $(\forall)$. Let $\mathcal{M}$ be the first-order structure that corresponds to Γ, i.e. the universe $|\mathcal{M}|$ is the set $[C]$ of equivalence classes of the constants in $abs(S^n)$, and $\mathcal{M}$ interprets the unary and binary predicate symbols, as well as the abstracted quantified formulas $\boxed{\varphi}$ that appear in $abs(S^n)$. We will prove that $\mathcal{M}$ is a model of S^0.

Step 1. We again prove that for every ground instance of every subformula ψ of a formula in S^0 for constants that appear in $abs(S^n)$, if $\mathcal{M} \models abs(\psi)$ then $\mathcal{M} \models \psi$.

- For literals ψ that are ground instances of subformulas in S^n (and a fortiori in S^0), we now have $abs(\psi) = \psi$, and therefore $\mathcal{M} \models abs(\psi)$ iff $\mathcal{M} \models \psi$. Note that this argument extends to ground instances of the new subformulas $x = i$ introduced by the translation from hybrid logic.
- For conjunctions and disjunctions, the proof of the inductive step is immediate.
- For ground instances ψ of quantified formulas of the forms $\exists y : R(c,y) \wedge \varphi(y)$ and $\forall y : \neg R(c,y) \vee \varphi(y)$, the arguments are exactly the same as in the proof of theorem 5, replacing $\boxed{\psi} \in \Gamma$ by $\mathcal{M} \models \boxed{\psi}$.

Step 2. For any formula $\varphi \in S^0$, we have that φ is ground and $\mathcal{M} \models abs(\varphi)$ because $\mathcal{M}$ is a model of $abs(S^n) \supseteq abs(S^0)$. By step 1, we conclude that $\mathcal{M} \models \varphi$. □

Termination. The termination proof follows exactly the lines of that of Theorem 6, except that now S^0 may contain several constants, corresponding to nominals and to the Skolem constant for the world from which the model construction for S^0 starts. All these (finitely many) constants are assigned depth 0. The procedure applies the same rules as for the case of basic modal logic, and the structure of the formulas is essentially the same (up to the addition of atomic equalities), and therefore termination is ensured by the same argument.

The fact that the proofs of soundness, completeness, and termination carry over in a straightforward way from basic modal logic to hybrid logic is in marked contrast to the situation for tableau calculi. In particular, the proof of termination is highly non-trivial for tableaux for hybrid logic [6].

5 Conclusions and Related Work

We have presented an SMT-based decision procedure for modal logic and some of its extensions. It is based on combining a ground solver for propositional logic or quantifier-free first-order logic with a custom instantiation module that lazily produces instances of quantified formulas as directed by the ground solver. The procedure robustly extends from basic modal logic to hybrid logic, the main difference being the replacement of a SAT solver by an SMT solver for reasoning about equality. We have also adapted the procedure to take into account standard conditions on the accessibility relation, such as reflexivity, symmetry, and transitivity; for lack of space, these extensions will be described elsewhere. Further extensions, such as to the guarded fragment of predicate logic [1] that generalizes modal logic while retaining its good computational properties, are an interesting avenue for future work.

In principle, the instantiation procedure can be implemented using patterns and e-matching, which are provided by standard SMT solvers. However, one should not expect such a naive implementation to yield an efficient decision procedure for modal logics. In particular, without imposing further control, subsequent calls to the ground solver may result in completely different ground models. One way to obtain a decision

procedure running in polynomial space is to impose a depth-first search strategy similar to modal tableaux, where instances corresponding to fully explored branches can be forgotten for the remainder of the exploration. This will imply for the SMT solver to forget instances in a smart way, and such a feature requires some careful engineering. Our preliminary experiments confirm that SMT solvers with trigger-based instantiation as currently implemented are off-the-shelf decision procedures for basic modal logic, but also show that forgetting instances is a required first step towards efficiency. More efficient translations from modal to first-order logic, such as the functional translation and its variants [19], may also help improving the performance without changing the basic setup of our procedure.

Whereas semantic tableaux remain state of the art decision procedures for modal and related logics, several authors proposed alternative methods. In particular, Hustadt and Schmidt [10,16,21] systematically explored techniques based on translations to first-order logic and the use of resolution and superposition provers. Our work starts from the same encodings in first-order languages but relies on ground decision procedures for suitable fragments of first-order logic. We believe that our proofs are more elementary, and we hope to obtain similarly efficient implementations by controlling repeated calls to the ground solver.

Sebastiani et al. [11,24] investigate the use of SAT solvers for modal and related logics. Their approach does not start from an encoding into first-order logic, but abstracts formulas with top-level modal connectives, similar to our abstractions of quantified formulas. When the set of abstracted formulas is found satisfiable, the SAT solver is launched again on sets of sub-problems derived from the formulas beneath the topmost modal operators. Whereas their decision procedure is very efficient for basic modal logic, it appears to fundamentally depend on the clean separation of truth conditions for worlds that correspond to distinct modal depths, and this condition is not satisfied for many extensions of modal logic, including hybrid logic.

Our small investigation into SMT-based decision procedures for modal and related logics owes in part to Martin Wirsing's interest in modal logic, as witnessed by Chapter 6 of [4]. Our work is being developed within an ongoing cooperation between teams located in Argentina and in Europe, which Martin actively fostered. For both reasons, we hope that this paper is a suitable contribution for the present volume, and we present Martin our sincere wishes for many more years of intellectual happiness in pursuing research at the interface of algebra, logic, and computer science.

References

1. H. Andréka, J. van Benthem, and I. Németi. Modal languages and bounded fragments of predicate logic. *Journal of Philosophical Logic*, 27:217–274, 1998.
2. C. Areces and B. ten Cate. Hybrid logics. In P. Blackburn, F. Wolter, and J. van Benthem, editors, *Handbook of Modal Logics*, pages 821–868. Elsevier, 2006.
3. C. Barrett, R. Sebastiani, S. A. Seshia, and C. Tinelli. Satisfiability modulo theories. In A. Biere, M. J. H. Heule, H. van Maaren, and T. Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, chapter 26, pages 825–885. IOS Press, Feb. 2009.

4. F. L. Bauer and M. Wirsing. *Elementare Aussagenlogik*. Springer, Berlin Heidelberg, 1991.
5. P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*, volume 53 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, 2001.
6. T. Bolander and P. Blackburn. Termination for hybrid tableaus. *Journal of Logic and Computation*, 17(3):517–554, 2007.
7. E. Börger, E. Grädel, and Y. Gurevich. *The Classical Decision Problem*. Springer-Verlag, Berlin, 1997. With an appendix by Cyril Allauzen and Bruno Durand.
8. A. Chandra and P. Merlin. Optimal implementation of conjunctive queries in relational databases. In *Proc. 9th ACM Symp. Theory of Computing*, pages 77–90, 1977.
9. A. Church. A note on the Entscheidungsproblem. *Journal of Symbolic Logic*, 1:40–41, 1936.
10. H. Ganzinger, U. Hustadt, C. Meyer, and R. A. Schmidt. A resolution-based decision procedure for extensions of K4. In M. Zakharyaschev, K. Segerberg, M. de Rijke, and H. Wansing, editors, *Advances in Modal Logic*, pages 225–246. CSLI Publications, 1998.
11. F. Giunchiglia and R. Sebastiani. Building decision procedures for modal logics from propositional decision procedures: The case study of modal K(m). *Information and Computation*, 162(1-2):158–178, 2000.
12. V. Goranko and S. Passy. Using the universal modality: Gains and questions. *Journal of Logic and Computation*, 2(1):5–30, 1992.
13. E. Grädel. Why are modal logics so robustly decidable? *Bulletin EATCS*, 68:90–103, 1999.
14. E. Grädel, P. Kolaitis, and M. Vardi. On the decision problem for two-variable first-order logic. *Bulletin of Symbolic Logic*, 3:53–69, 1997.
15. E. Grädel, M. Otto, and E. Rosen. Two-variable logic with counting is decidable. In *Proc. 12th Ann. IEEE Symp. Logic in Computer Science (LICS 1997)*, pages 306–317. IEEE Comp. Soc., 1997.
16. U. Hustadt, H. de Nivelle, and R. A. Schmidt. Resolution-based methods for modal logics. *Logic Journal of the IGPL*, 8(3):265–292, 2000.
17. L. Löwenheim. Über Möglichkeiten im Relativkalkül. *Mathematische Annalen*, 76:447–470, 1915.
18. R. Nieuwenhuis, A. Oliveras, and C. Tinelli. Solving SAT and SAT Modulo Theories: From an abstract Davis-Putnam-Logemann-Loveland procedure to DPLL(T). *J. ACM*, 53(6):937–977, 2006.
19. H. J. Ohlbach and R. A. Schmidt. Functional translation and second-order frame properties of modal logics. *Journal of Logic and Computation*, 7(5):581–603, 1997.
20. L. Pacholsky, W. Szwast, and L. Tendera. Complexity of two-variable logic with counting. In *Proc. 12th Ann. IEEE Symp. Logic in Computer Science (LICS 1997)*, pages 318–327, 1997.
21. R. A. Schmidt and U. Hustadt. First-order resolution methods for modal logics. In A. Voronkov and C. Weidenbach, editors, *Programming Logics – Essays in Memory of Harald Ganzinger*, volume 7797 of *Lecture Notes in Computer Science*, pages 345–391. Springer, 2013.
22. R. A. Schmidt and D. Tishkovsky. Using tableau to decide description logics with full role negation and identity. *ACM Trans. Comput. Log.*, 15(1), 2014.
23. D. Scott. A decision method for validity of sentences in two variables. *Journal of Symbolic Logic*, 27(377):74, 1962.
24. R. Sebastiani and A. Tacchella. SAT techniques for modal and description logics. In A. Biere, M. Heule, H. van Maaren, and T. Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 781–824. IOS Press, 2009.
25. A. Turing. On computable numbers, with an application to the 'Entscheidungsproblem'. *Proc. London Mathematical Society 2nd. series*, 42:230–265, 1937.
26. M. Vardi. Why is modal logic so robustly decidable? In *DIMACS Ser. Disc. Math. Theoret. Comp. Sci. 31*, pages 149–184. AMS, 1997.