Deciding Modal Logics using Tableaux and Set Theory

Carla Piazza and Alberto Policriti

Abstract

We propose a tableau-like decision procedure for deciding the satisfiability of set-theoretical formulae with restricted universal quantifiers and the powerset operator. Our result apply to a rather large class of set theories. The procedure we define can be used as a subroutine to decide the same class of formulae both in Set Theory and in non well-founded set theories, since we assume neither Regularity nor any form of anti-foundation axiom. Moreover, the decidability result presented allow to characterize a class of decidable modal logics. Thanks to the $\Box$-as-$\mathcal{P}$ (box-as-powerset) translation our procedure can be used to uniformly study a large class of modal logics which includes K, T, $S4$, $S5$, $S4.3$.

1 Introduction

Since the late 1970's decidability results for fragments of Set Theory have been studied [13] with the long-term aim of providing a collection of procedures capable of computing within the *decidable core* of Set Theory [10]. Two of the main features of Set Theory are the Extensionality and the Regularity axioms. The first establishes a strong link between membership and equality, while the second implies that the membership relation forms no cycles or infinite descending chains. In the 1980's the necessity to consider theories non assuming these strong constraints (re-)emerged in many communities, hence various proposals for (axiomatic) non well-founded set theories (and universes) were developed (see [14, 1, 2, 3]). In general, as far as the choice of the underlying axioms is concerned, when dealing with decidability problems the safest (and often most reasonable) choice is a minimal one. Once a decision procedure over a minimal theory is discovered one can usually tune it up in order to deal with stronger theories (cf. [15]).

The flexibility provided by the absence of some axioms allows also to play with sets in a rather peculiar way: in [12, 5, 6], exploiting the possibilities given by a weak set theory, a set-theoretic translation—called $\Box$-as-$\mathcal{P}$ (box-as-powerset) translation—rewriting any modal formula into a set-theoretic one, was proposed. Such a method shows that, with the limited amount of second-order logic introduced by means of

Dip. di Matematica e Informatica, Univ. di Udine. Via Le Scienze 206, 33100 Udine (Italy). (piazza|policrit)@dimi.uniud.it

the axiomatic set theory driving the translation, a mild form of *deduction theorem* holds for a modal logic specified by any set of Hilbert's axioms. Even tough the proposed translation method was originally conceived as a way to automatize modal deduction, it can also be used as a tool for proving decidability of classes of modal logics (cf. [11]). The basic idea is that in the case of a first-order complete modal logic it is possible to replace the $\Box$-as-$\mathcal{P}$ translation of the axioms with the first-order characterization of the modal logic. In (cf. [11]) is presented a way of reducing modal decidability problems to set-theoretic ones. A conjecture stated at the end of that paper has been our starting point for this work, for which a tableau-based approach to the decidability problem turned out to be the right way to approach the question.

In order to provide decision procedures for a set-theoretic class of formulae obtained through the above mentioned translation technique, we propose here a tableau-based procedure to decide satisfiability w.r.t. the minimal set theory Ω. The decidability result presented here allows to cope with the powerset operator, together with the usual operators of MLS, Multi-Level Syllogistic, and restricted-universal quantifiers, in models of set theories that are neither well-founded nor extensional. Similar decidability problems have already been treated in the well-founded case,with a tableau-based decision procedure, in [7, 8, 9]. In [17] we presented a tableau method which works in the non-well-founded case. There are many differences between the problem we consider here and the problem studied in [17]: in [17] we did not allow universal quantifiers, we did not put restrictions on the use of the powerset operator, and we used the anti-foundation axiom (AFA); here we allow universal quantifiers, we put restrictions on the use of the powerset operator, considering only the formulae with powerset which could be obtained using the $\Box$-as-$\mathcal{P}$ translation, and we do not assume (AFA) or any other form of anti-foundation.

The paper is organized as follows. In Section 2 we introduce the theory Ω, and we recall the $\Box$-as-$\mathcal{P}$ translation. In Section 3 we present a construction which builds models of Ω from graphs: the construction is interesting because it works as our satisfiability procedure, but it uses the axioms of Ω instead of an input formula. In Section 4, extending a result presented in [11], we motivate the satisfiability problem we deal with. In Section 5 we define our tableau satisfiability procedure, while in Section 6 we introduce a termination condition on our procedure and we show its correctness and completeness w.r.t. a subclass of formulae. In Section 7 we analyze the modal meaning of our result. Some conclusions and future works are presented in Section 8. The proofs of the results in this paper can be found in [16].

2 Preliminaries and Motivations

The basic modal logic notions and the relationships between modal logic and first-order logic through the language $\{R, =\}$ we refer to in this paper, can be found in [4]. Consider the first-order language $\mathcal{L} = \langle \mathcal{F}, \mathcal{P} \rangle$, with $\mathcal{F} = \{\cup, \setminus, \wp\}$ and $\mathcal{P} = \{\in, \subseteq\}$. If φ is a formula, $FV(\varphi)$ stands for the free variables in φ.

Let Ω be the theory in Figure 1.

Definition 2.1 *If $\gamma(p_1, \ldots, p_n)$ is a modal formula, the set-term $\gamma^*(x, z_1, \ldots, z_n)$ is inductively defined as follows:*

$(\cup)$	$\forall x,\, y,\, z\,(x \in y \cup z \leftrightarrow x \in y \vee x \in z)$
$(\setminus)$	$\forall x,\, y,\, z\,(x \in y \setminus z \leftrightarrow x \in y \wedge x \notin z)$
$(\subseteq)$	$\forall x,\, y\,(x \subseteq y \leftrightarrow \forall z\,(z \in x \rightarrow z \in y))$
$(\wp)$	$\forall x,\, y\,(x \in \wp(y) \leftrightarrow x \subseteq y)$

Figure 1: The theory Ω.

1. $p_i^* = z_i$;
2. $(\gamma_1 \vee \gamma_2)^* = \gamma_1^* \cup \gamma_2^*$;
3. $(\neg\gamma)^* = x \setminus \gamma^*$;
4. $(\Box\gamma)^* = \wp(\gamma^*)$.
The set-theoretic translation $\gamma^s(x)$ *of the formula* $\gamma(p_1, \ldots, p_n)$ *is the formula*

$$\forall z_1, \ldots, z_n(x \subseteq \gamma^*(x, z_1, \ldots, z_n)).$$

One of the results in [12] is that if Λ is the complete modal logic obtained by adding the axiom Ax to the modal logic K (see [4]), then

$$\vdash_\Lambda \gamma \quad \text{iff} \quad \Omega \vdash \forall x(x \subseteq \wp(x) \wedge Ax^s(x) \rightarrow \gamma^s(x)).$$

One of the main feature of the above translation ($\Box$-as-$\mathcal{P}$ translation) is its ability to deal with any modal logic, regardless the fact that such a logic is first-order definable or not. On the other hand, the results presented in [11] were direct to a study of the possibilities of transferring decidability results to and from Modal Logic and Set Theory and first-order complete modal logic are taken into account. Λ is a first-order complete modal logic (which extends K) with first-order correspondent δ in the language $\{R, =\}$ (see [4]) if it holds that

$$\vdash_\Lambda \gamma \qquad \text{iff} \qquad \gamma \text{ is valid in every frame in which } \delta \text{ is valid.}$$

Notice that δ is a sentence (i.e., the set of free variables in δ is empty, $FV(\delta) = \emptyset$). It is possible to translate δ into a set-theoretic formula using the following definition.

Definition 2.2 *Let δ be a first-order formula in the language $\{R, =\}$, and let x be a variable which does not occur in δ. The formula $\delta^s(x)$ is the formula we obtain from δ by:*
1. replacing all the occurrences of R with $\in$;
2. replacing all the $\forall y$ and all the $\exists y$ with $\forall y \in x$ and $\exists y \in x$ respectively.

Using a suitable extension Ω' of Ω (that is $\Omega' = \Omega + Cc + s$, cf. [11]) it has been proved that if Λ is a first-order complete modal logic (which extend K) with first-order correspondent δ, then

$$\vdash_\Lambda \gamma(p_1, \ldots, p_n) \quad \text{iff} \quad \Omega' \vdash \forall x(x \subseteq \wp(x) \wedge \delta^s(x) \rightarrow \gamma^s(x)). \tag{1}$$

Spelling out all the variables and the meaning of the formulae, introduced in the translation, we have that if the class of formulae of the form

$$\forall x \forall \vec{z} \forall \vec{v}(x \subseteq \wp(x) \wedge \forall \vec{y} \in x(\alpha^s(x, \vec{y}, \vec{v})) \rightarrow x \subseteq \gamma^*(x, \vec{z})), \tag{2}$$

where α is a quantifier-free conjunction of clauses over the language $\{R, =\}$, were decidable over Ω', then all first-order complete modal logics whose first-order correspondent is in the class $\exists^*\forall^*$ (formulae in prenex form with an arbitrary number of $\exists$ followed by an arbitrary number of $\forall$) would be decidable.

In the following sections we prove that in (1), and hence in (2), it is possible to replace Ω' with the much more elementary Ω. Then we present a procedure which deals with this last decidability problem. We prove that when δ is in $\exists^*\forall^2$ and α does not contain equalities we are able to decide the satisfiability problem adding a termination condition. In this way we obtain that using a *unique* decision procedure, a sort of uniform method to build canonical counter-examples, we can uniformly decide over a large class of modal logics.

3 From Graphs to Models of Ω

In order to build a model of Ω in which an unquantified formula ψ is satisfiable, one natural idea is to build a graph whose nodes are the variables in ψ and whose edges mimic the membership atoms of ψ ([17]). Then we need to embed this graph into a model of Ω. In this section we define how to build a model of Ω out of a graph and during the construction we need to *saturate* the graph in order to satisfy the axioms of Ω. The procedure Check (see Section 5) in a certain sense generalizes such kind of construction dealing with formulae in the class we want to decide. We will use the result we give in this section both to extend to Ω the validity of (1) above and to prove the correctness of our procedure Check.

Let $G = \langle N, E\rangle$ be a graph, with N possibly infinite. We define a method which allows us to map G into a model G_ω of Ω such that G is a sub-graph of G_ω, *without adding out-going edges to the nodes of* G (i.e., maintaining G as a complete subgraph of G_ω).

First, we introduce some notations. Let $G = \langle N, E\rangle$, let $v, u \in N$ be two nodes, and let $V \subseteq N$ be a set of nodes:
- $\mathsf{in}(G, v, u, \cup)$ iff there exists a node w in N such that
  $\forall x \in N((\langle v, x\rangle \in E \vee \langle u, x\rangle \in E) \leftrightarrow \langle w, x\rangle \in E)$;
- $\mathsf{in}(G, v, u, \setminus)$ iff there exists a node w in N such that
  $\forall x \in N((\langle v, x\rangle \in E \wedge \langle u, x\rangle \notin E) \leftrightarrow \langle w, x\rangle \in E)$;
- $\mathsf{sb}(G, V, v)$ iff $\forall x \in N(x \in V \rightarrow \langle v, x\rangle \in E)$;
- $\mathsf{in}(G, V, v, \subseteq)$ iff there exists a node w in N such that
  $\forall x \in N((x \in V \wedge \langle v, x\rangle \in E) \leftrightarrow \langle w, x\rangle \in E)$;
- $\mathsf{csb}(G, V, v)$ iff $\neg\mathsf{sb}(G, V, v) \vee \mathsf{in}(G, V, v, \subseteq)$;
- $\mathsf{in}(G, v, \wp)$ iff there exists a node w in N such that
  $\forall x \in N(\forall y \in N(\langle x, y\rangle \in E \rightarrow \langle v, y\rangle \in E) \leftrightarrow \langle w, x\rangle \in E)$.

Intuitively $\mathsf{in}(G, v, u, \cup)$ means that there is a node in G which can be used has $v \cup u$ (i.e., it satisfies the axiom $(\cup)$ w.r.t. v and u). Since $x \in v$ is interpreted on the graph has $\langle v, x\rangle$, $\mathsf{sb}(G, V, v)$ means that V is a subset of v, while $\mathsf{in}(G, V, v, \subseteq)$ stands for the fact that there is a node in G which can represent the subset $V \cap v$ of v. It follows that not $\mathsf{csb}(G, V, v)$ is equivalent to the fact that V is a subset of v and there is no node in G which represents this subset.

Starting from $G = G_0$ we introduce a sequence of graphs G_n and then we show

that the graph obtained as the union of all the graphs in the sequence is the model of Ω we are looking for.
If we have defined G_n, with $n = 4m$ we inductively define $G_{n+i} = \langle N_{n+i}, E_{n+i}\rangle$, for $i = 1, 2, 3, 4$, as follows:

$$\begin{array}{rcl} N_{n+1} & = & N_n \cup \{\langle v, u, \cup\rangle \mid v, u \in N_n,\ \text{not } \mathsf{in}(G_n, v, u, \cup)\} \\ E_{n+1} & = & E_n \cup \{\langle\langle v, u, \cup\rangle, x\rangle \mid \langle v, u, \cup\rangle \notin N_n, \langle v, x\rangle \in E_n \text{ or } \langle u, x\rangle \in E_n\} \end{array}$$

$$\begin{array}{rcl} N_{n+2} & = & N_{n+1} \cup \{\langle v, u, \setminus\rangle \mid v, u \in N_{n+1},\ \text{not } \mathsf{in}(G_{n+1}, v, u, \setminus)\} \\ E_{n+2} & = & E_{n+1} \cup \{\langle\langle v, u, \setminus\rangle, x\rangle \mid \langle v, u, \setminus\rangle \notin N_{n+1}, \langle v, x\rangle \in E_{n+1}, \langle u, x\rangle \notin E_{n+1}\} \end{array}$$

$$\begin{array}{rcl} N_{n+3} & = & N_{n+2} \cup \{\langle v, V, \subseteq\rangle \mid \{v\}, V \subseteq N_{n+2},\ \text{not } \mathsf{csb}(G_{n+2}, V, v)\} \\ E_{n+3} & = & E_{n+2} \cup \{\langle\langle v, V, \subseteq\rangle, x\rangle \mid \langle v, V, \subseteq\rangle \notin N_{n+2}, x \in V\} \end{array}$$

$$\begin{array}{rcl} N_{n+4} & = & N_{n+3} \cup \{\langle v, \wp\rangle \mid v \in N_{n+3},\ \text{not } \mathsf{in}(G_{n+3}, v, \wp)\} \\ E_{n+4)} & = & E_{n+3} \cup \{\langle\langle v, \wp\rangle, x\rangle \mid \langle v, \wp\rangle \notin N_{n+3},\ \mathsf{sb}(G_{n+3}, \{y \mid \langle x, y\rangle \in E_{n+3}\}, v)\} \end{array}$$

Let $G_\omega = \langle N_\omega, E_\omega\rangle$, with $N_\omega = \bigcup_{n\in\mathbb{N}} N_n$ and $E_\omega = \bigcup_{n\in\mathbb{N}} E_n$

Let τ be a well-ordering over N_ω. From G_ω we obtain an interpretation of $\mathcal{L}$ defining $\forall v_1, v_2, \in N_\omega$:
- $u \in v$ if and only if $\langle v, u\rangle \in E_\omega$;
- $v \cup u = w$, with w the min (w.r.t. τ) such that $\forall x(x \in w \leftrightarrow x \in v \vee x \in u)$;
- $v \setminus u = w$, with w the min (w.r.t. τ) such that $\forall x(x \in w \leftrightarrow x \in v \wedge x \notin u)$;
- $u \subseteq v$ if and only if $\forall x(x \in u \rightarrow x \in v))$;
- $\wp(v) = w$, with w the min (w.r.t. τ) such that $\forall x(x \in w \leftrightarrow x \subseteq v)$.

We use G_ω to refer to this interpretation of $\mathcal{L}$, forgetting that it is τ-dependent. We have to prove that G_ω is a model of Ω and that G is a complete subgraph in G_ω. First we prove that this last property holds for all the G_m, hence we use this fact to prove that G_ω is a model of Ω.

Lemma 3.1 *For all $m \in \mathbb{N}$, G_m is a complete subgraph in G_ω*

Another important property of our construction is that we never add a node whose set of outgoing edges is equal to the set of outgoing edges of a node which was already in the graph at the previous iteration.

Lemma 3.2 *Let u and v be two nodes in G_ω, such that $\{x \mid \langle v, x\rangle \in E_\omega\} = \{x \mid \langle u, x\rangle \in E_\omega\}$. If $u \in E_m$, then $v \in E_m$.*

Lemma 3.3 *G_ω is an interpretation of $\mathcal{L}$, and a model of Ω.*

4 Ω and Modal Logic

In Section 2 we recalled the definition of the $\Box$-as-$\mathcal{P}$ translation (see [5]). Here we precisely extend to Ω the result (1) at the ground of the technique for transferring set-theoretic decidability results to Modal Logic. In particular, we introduce the specific satisfiability problem we deal with in the rest of the paper.

Let Λ be a first-order complete modal logic (which extends K) with first-order correspondent δ in the language $\{R, =\}$.

Proposition 4.1 *For any modal formula $\gamma(p_1, \ldots, p_n)$ we have that*

$$\vdash_\Lambda \gamma(p_1, \ldots, p_n) \quad \textit{iff} \quad \Omega \vdash \forall x(x \subseteq \wp(x) \wedge \delta^s(x) \rightarrow \forall \vec{z}(x \subseteq \gamma^*(x, z_1, \ldots, z_n))).$$

Now if we want to decide

$$\vdash_\Lambda \gamma,$$

from Proposition 4.1 it follows that we can equivalently decide

$$\Omega \vdash \forall x(x \subseteq \wp(x) \wedge \delta^s(x) \rightarrow \forall \vec{z}(x \subseteq \gamma^*(x, z_1, \ldots, z_n))).$$

This is equivalent to decide the unsatisfiability in Ω of

$$\neg(\forall x(x \subseteq \wp(x) \wedge \delta^s(x) \rightarrow \forall \vec{z}(x \subseteq \gamma^*(x, z_1, \ldots, z_n)))).$$

Bringing this formula towards its prenex normal form, we obtain that we can equivalently decide the unsatisfiability in Ω of

$$\exists x \exists \vec{z} \exists w(x \subseteq \wp(x) \wedge \delta^s(x) \wedge w \in x \wedge w \notin \gamma^*(x, z_1, \ldots, z_n)).$$

When δ is of the form $\exists^* \forall^* \alpha$ we can give to the above formula the equivalent form

$$\exists x \exists \vec{z} \exists w \exists \vec{v}(x \subseteq \wp(x) \wedge \forall \vec{y} \in x(\alpha^s(x, \vec{y}, \vec{v})) \wedge w \in x \wedge w \notin \gamma^*(x, z_1, \ldots, z_n)).$$

The above argument shows that:

Proposition 4.2 *If the satisfiability problem w.r.t. Ω of the class of formulae of the form*[1]

$$\exists x \exists \vec{z} \exists w \exists \vec{v}(x \subseteq \wp(x) \wedge \forall \vec{y} \in x(\alpha^s(x, \vec{y}, \vec{v})) \wedge w \in x \wedge w \notin \gamma^*(x, z_1, \ldots, z_n)),$$

where α^* is quantifier-free, is decidable, then the decidability of all the first-order complete modal logics with first-order correspondent in the class $\exists^* \forall^*$ follows.

In the next two sections we study the above satisfiability problem. In the general case we are able to give a procedure which could not terminate. In the case δ is in the class $\exists^* \forall^2$ and in α^s there are no positive equality literals we are able to add a test to the procedure in order to obtain a procedure which decide the satisfiability problem.

5 The Tableau Satisfiability Procedure

As we explained in the previous section, the class of formulae we are interested in is the class $\mathcal{C}_1$ of formulae of the form

$$x \subseteq \wp(x) \wedge \varphi_1 \wedge \ldots \wedge \varphi_n \wedge w \in x \wedge w \notin t(x, \vec{z})$$

where

$$\varphi_i \equiv \forall \vec{y} \in x(p^i_1(\vec{y}, \vec{v}) \vee \ldots \vee p^i_{m_i}(\vec{y}, \vec{v})) \tag{3}$$

[1]A formula F is satisfiable w.r.t. a theory $\mathbb{T}$ if and only if there exists a model of $\mathbb{T}$ in which F is satisfiable.

with $p^i_j(\vec{y})$ in one of the following forms

$$\begin{array}{c} y_h \in y_k,\ y_h \notin y_k,\ y_h \neq y_k, y_h = y_k \\ y_h \in v_k,\ y_h \notin v_k,\ y_h \neq v_k, y_h = v_k \\ v_h \in y_k,\ v_h \notin y_k,\ v_h \neq y_k, v_h = y_k \end{array}$$

and

$$t(x,\vec{z}) ::= z_j \,|\, x \,|\, t_1 \cup t_2 \,|\, x \setminus t_1 \,|\, \wp(t_1). \tag{4}$$

We want to test if there exists at least one model of Ω in which a formula ψ of this class is satisfiable. Notice that $FV(\psi) = \{x, w, \vec{v}, \vec{z}\}$. Let us recall the connection between this satisfiability problem and the problem $\vdash_\Lambda \gamma$, when Λ has δ as first order correspondent:

- the variable x represent the *frame* we are looking for, i.e. a frame in which δ is valid and γ is not;
- $\varphi_1 \wedge \ldots \wedge \varphi_n$ is our δ^s;
- the variables y's and v's are *worlds* of our frame x;
- the variable w represent the world in which we want to *falsify* γ;
- $t(x,\vec{z})$ is the $\Box$-as-$\mathcal{P}$ translation of γ, i.e. it is *the set of world at which* γ *is true*;
- the z's represent the *values* (set of worlds) we must assign to the propositional letters in γ in order to make that w falsify γ.

The main idea behind our decision procedure is the following: we build the frame (set) x starting from its world (element) w; at each iteration we add the new worlds (elements) that are necessary to satisfy δ and to ensure that w does not satisfy γ (w does not belong to γ^*).

The fact that we look at the problem from a set-theoretic perspective gives us the advantage that:

- in order to add the new elements to x (and to the subterms of $t(x,\vec{z})$, when it is necessary), we can use the rules we obtain from the natural meaning that Ω gives to the set-operators;
- in order to ensure termination it is sufficient to check to which subsets of x we are adding elements: if these subsets are already "big enough" we can stop our procedure.

In order to decide whether a formula in the class $\mathcal{C}_1$ is satisfiable it is convenient to deal with a larger class $\mathcal{C}$ of formulae.

Definition 5.1 *Let $\{x, z_1, \ldots, z_p, v_1, \ldots, v_q\} (= \{x, \vec{z}, \vec{v}\})$ be a finite set of distinct variables, and $Term = \{t_1(x,\vec{z}), \ldots, t_s(x,\vec{z})\}$ be a finite set of terms of the form (4). Let $A, B \subseteq I_q \times I_s$ and $C, D, E \subseteq I_q \times I_q$.*[2] *A formula $\psi(x,\vec{z},\vec{v})$ is in the class $\mathcal{C}$ if and only if it is of the form*

$$\psi_\forall(x,\vec{v}) \wedge x \subseteq \wp(x) \wedge \psi_{\mathsf{term}}(x,\vec{z},\vec{v}) \wedge \psi_{\mathsf{var}}(\vec{v}),$$

[2] Let $n \in \mathbb{N}, n > 0$, we use I_n to refer to $\{1, \ldots, n\}$.

where

$$\psi_{\forall}(x,\vec{v}) \equiv \bigwedge_{i=1}^{n} \varphi_i,$$

with the φ_i's of the form (3),

$$\psi_{\mathsf{term}}(x,\vec{z},\vec{v}) \equiv \bigwedge_{\langle a_1,a_2\rangle \in A} (v_{a_1} \notin t_{a_2}(x,\vec{z})) \wedge \bigwedge_{\langle b_1,b_2\rangle \in B} (v_{b_1} \in t_{b_2}(x,\vec{z})),$$

and

$$\psi_{\mathsf{var}} \equiv \bigwedge_{\langle c_1,c_2\rangle \in C} (v_{c_1} \notin v_{c_2}) \wedge \bigwedge_{\langle d_1,d_2\rangle \in D} (v_{d_1} \in v_{d_2}) \wedge \bigwedge_{\langle e_1,e_2\rangle \in E} (v_{e_1} \neq v_{e_2}).$$

The fact that in ψ_{var} we do not have equalities is not a restriction: it is possible to map a formula in which there are equalities of the form $v_h = v_k$ into an equivalent formula in $\mathcal{C}$. The class $\mathcal{C}_1$ is a subclass of $\mathcal{C}$. A formula in $\mathcal{C}$ still have a modal interpretation: we are looking for a frame x whose accessibility relation satisfies certain conditions ($\psi_{\forall} \wedge \psi_{\mathsf{var}}$) and in which there are worlds which do not satisfy and do satisfy certain modal formulae (ψ_{term}).

We now start to describe a procedure to test the satisfiability of formulae of $\mathcal{C}$ w.r.t. the theory Ω. Let us consider the set of rules, $\exists$-reductions, in Figure 2. The notation used in rules $(2_\exists)$ and $(5_\exists)$ means that it is necessary to choose

$$\frac{w \notin s_1 \cup s_2}{w \notin s_1 \wedge w \notin s_2}\ (1_\exists) \qquad \frac{w \notin k \setminus s_1}{w \notin k \mid w \in s_1}\ (2_\exists) \qquad \frac{w \notin \wp(s_1)}{w_1 \in w \wedge w_1 \notin s_1}\ (3_\exists)$$

$$\frac{w_1 \in w \quad w \in k \quad k \subseteq \wp(k)}{w_1 \in k}\ (4_\exists)$$

$$\frac{w \in s_1 \cup s_2}{w \in s_1 \mid w \in s_2}\ (5_\exists) \qquad \frac{w \in k \setminus s_1}{w \in k \wedge w \notin s_1}\ (6_\exists) \qquad \frac{w_1 \in w \quad w \in \wp(s_1)}{w_1 \in s_1}\ (7_\exists)$$

Figure 2: The rules $\exists$-reductions.

(non-deterministically) one out of two alternatives. The rule $(4_\exists)$ is needed only to deal with the inclusion $x \subseteq \wp(x)$, since in our class of formulae there are no other inclusions. Notice that rule $(3_\exists)$ is the only one which leads to the introduction of a new variable. In these rules we make use of the variables w, w_1, s_i, k. When we apply these rules to formulae in $\mathcal{C}$ we have that: w's, w_1's correspond to variables in $\vec{v}$; the variables s_i's correspond to terms of $Term$; the variable k corresponds to x. Later we will use variables $\vec{h}$, which for formulae in $\mathcal{C}$ correspond to the variables $\vec{y}$.

Let $\frac{l}{r}$ be one of rules in $\exists$-reductions. Applying once this rule to a formula ψ in the class $\mathcal{C}$ consists in:

1. check if there exists a subformula ℓ in $\psi_{\mathsf{term}} \wedge x \subseteq \wp(x)$ and a substitution σ such that $l[\sigma] = \ell$;

2. check if $r[\sigma]$ is already a subformula of $\psi_{\mathsf{term}} \wedge \psi_{\mathsf{var}}$ and, if this is not the case, map ψ into $\psi \wedge r[\sigma]$.

Intuitively, what we want to do is to apply all the rules of ∃-reductions, until a fix-point is reached. Rule $(3_\exists)$ can cause a problem since it introduces new variables. However, in ψ there are only a finite number of $\wp$ operators and, moreover, the r.h.s. of $(3_\exists)$ is of 'lower complexity' than its l.h.s.; hence the fix-point will always be reached in a finite number of iterations.

Our next problem is that we must apply the ∃-reductions rules together with other rules we will introduce later in this section. It is an uncontrolled interaction of the ∃-reductions with these other rules that can lead to non-termination. This is why we introduce a procedure in which we control the application of the ∃-reductions. The procedure apply-∃ (Figure 3) takes as input a formula ψ in the class $\mathcal{C}$, a variable v, and a term t, and allows to apply rule $(3_\exists)$ only if $\sigma(w) = v$ and $\sigma(\wp(s_1)) = t$. In Figure 4 we give the second set of rules we need and by the procedure apply-∀

```
apply-∃($\psi, v, t$):
repeat
    $\varphi := \top$;
    for all $\ell$ subformulae of $\psi$ do
        if there is $i \in \{1_\exists, 2_\exists, 4_\exists, 5_\exists, 6_\exists, 7_\exists\}$ and $\sigma$ such that $\ell = l_i[\sigma]$ then
            if $r_i[\sigma]$ is not subformula of $\psi$ then
                $\psi := \psi \wedge r_i[\sigma]$;
                $\varphi := \bot$;
        if $\ell = v \notin t$ and $t = \wp(t_1)$ then
            if there is not $v_1$ such that $(v_1 \in v \wedge v_1 \notin t_1)$ is subformula of $\psi$ then
                $\psi := \psi \wedge v_1 \in v \wedge v_1 \notin t_1$;
                $\varphi := \bot$;
until $\varphi$.
```

Figure 3: The Procedure apply-∃.

(Figure 5) we explain how we control the application of these rules. apply-∀ takes a

$$\frac{\vec{w} \in k \quad \forall \vec{h} \in k(p_1 \vee \ldots \vee p_m)}{p_1[\vec{h}/\vec{w}] \ \mid \ \ldots \ \mid p_m[\vec{h}/\vec{w}]} \ (1_\forall)$$

Figure 4: The rule ∀-reductions.

formula ψ as input and applies the rule in ∀-reductions until a fix-point is reached. It also performs a collapsing of the variables on which an equality has been introduced. It is clear that a sufficient large number of applications of ∀-reductions to a formula ψ always reaches a fix-point (immediate consequence of the fact that this rule never adds new variables). As we mentioned before, a problem with termination could arise when we mix ∃-reductions and ∀-reductions.

To conclude, we need to give the rules to close the branches of the tableau: such general-reductions are presented in Figure 6. and it is clear how to apply them to a formula ψ in $\mathcal{C}$. The procedure in Figure 7 gives the details. Notice that if ψ is a

```
apply-$\forall(\psi)$:
repeat
    $\varphi := \top$;
    for all $\forall y_1, \ldots, y_h \in x(p_1 \vee \ldots \vee p_m)$ subformulae of $\psi$ do
        for all $(v_1 \in x \wedge \ldots \wedge v_h \in x)$ subformulae of $\psi$ do
            if there is not $k \leq m$ such that $p_k[\vec{y}/\vec{v}]$ is subformula of $\psi$ then
                let $k \leq m$;
                $\psi := p_k[\vec{y}/\vec{v}] \wedge \psi$;
                $\varphi := \bot$;
until $\varphi$.
for all $(v_1 = v_2)$ subformulae of $\psi$ do
    $\psi := \psi[v_1/v_2]$;
```

Figure 5: The Procedure apply-$\forall$.

$$\frac{w_1 \in s_1 \quad w_1 \notin s_1}{\bot}\,(1_g) \qquad \frac{w_1 \neq w_1}{\bot}\,(2_g)$$

Figure 6: The rules general-reductions.

formula in $\mathcal{C}$, then also apply-$\exists(\psi, w, s)$, apply-$\forall(\psi)$, and apply-gen(ψ) are in $\mathcal{C}$.

We introduce an auxiliary function part, which takes as input three sets and returns a Boolean value. part(A, B, C) returns true if and only if A and B partition C into one or two blocks.

$$\text{part}(A, B, C) := (A \cap B = \emptyset \text{ and } A \cup B = C).$$

The procedure Check (Figure 8) combines all the subroutines introduced up to this point in our general tableau procedure. A declarative version of Check would simply apply all the rules until a fix-point is reached. Our version considers also whether we entered in an infinite branch or not. Let us assume that in ψ_{term} there are m terms of the form $t(x, \vec{z})$ and that the set T of all their subterms has cardinality p. For each variable v in $\vec{v}$ we can guess to which elements of T v has to belong (the set EC_v) and, hence, to which elements of T v does not belong (the set NC_v). From the point of view of Kripke semantics this guess correspond to guess the subformulae of γ which holds at v. The $\exists$-reductions rules infer, using EC_v and NC_v, the membership relations and together with the general-reductions rules, they check the consistency of EC_v. The $\forall$-reductions rules add the membership literals which are necessary if we want that x satisfies $\psi_\forall$. Modally this corresponds to the fact that these rules "adjust" the frame in such a way that δ is valid. We know that rule $(3_\exists)$ can be applied using v at most p times, hence we put p occurrences of v in the queue *new*. Its meaning in modal logic's terms is that since there are at most p occurrences of $\Box$ in γ, we need to consider at most p immediate successors of v. At each iteration of the repeat-loop at most a new variable u is added. We *initialize* the values EC_u and NC_u and we add p occurrences of u in the queue *new*; the $\langle v, i\rangle$ which is taken

```
apply-gen($\psi$):
for all $\ell$ subformulae of $\psi$ do
    if there is $i \in \{1,2\}$, and $\sigma$ such that $\ell = l_i[\sigma]$ then
        $\psi := \bot$.
```

Figure 7: The Procedure apply-gen.

```
Check($\psi(x, \vec{z}, \vec{v})$):
$V := \{v \mid v \in \vec{v}\}$;
$T := \{t' \mid t(x,\vec{z})$ is a term in $\psi_{\mathsf{term}}$ and $t'$ is a subterm of $t(x,\vec{z})$ or $t' \in V \cup \{x\}\}$;
$p := |T|$;
let $ord$ be an ordering on $T$;
$new := []$;
$old := []$;    % only for termination
$Inf := \bot$;    % only for termination
for all $v \in V$ do    % only for termination
    $E_v := \{t' \mid t' \in T$ and $v \in t'$ is a subformula of $\psi_{\mathsf{term}}\}$;
    $N_v := \{t' \mid t' \in T$ and $v \notin t'$ is a subformula of $\psi_{\mathsf{term}}\}$;
    let $\langle EC_v, NC_v\rangle$ such that
        $EC_v \supseteq E_v$ and $NC_v \subseteq N_v$ and part$(EC_v, NC_v, T)$;
    $\psi := \psi \wedge \bigwedge_{t' \in EC_v} v \in t' \wedge \bigwedge_{t' \in NC_v} v \notin t'$
    $new :=$ append$(new, [\langle v,1\rangle, \ldots, \langle v,p\rangle])$;
repeat
    $\varphi := \psi$;
    $\langle v, i\rangle :=$ head$(new)$;
    $new :=$ cons$(new)$;
    $old :=$ append$(old, [\langle v,i\rangle])$;    % only for termination
    apply-$\exists(\psi, v, ord(i))$;
    apply-$\forall(\psi)$;
    if $u \in FV(\psi) \setminus FV(\varphi)$ then    % only for termination
        $E_u := \{t' \mid t' \in T$ and $u \in t'$ is a subformula of $\psi_{\mathsf{term}}\}$;
        $N_u := \{t' \mid t' \in T$ and $u \notin t'$ is a subformula of $\psi_{\mathsf{term}}\}$;
        let $\langle EC_u, NC_u\rangle$ such that
            $EC_u \supseteq E_u$ and $NC_u \subseteq N_u$ and part$(EC_u, NC_u, T)$;
        $\psi := \psi \wedge \bigwedge_{t' \in EC_u} u \in t' \wedge \bigwedge_{t' \in NC_u} u \notin t'$
        $new :=$ append$(new, [\langle u,1\rangle, \ldots, \langle u,p\rangle])$;
    apply-gen$(\psi)$;
%   if $\{w \mid \langle w,j\rangle \in new\} \cap V = \emptyset$ then
%       if $\{EC_w \mid \langle w,j\rangle \in old\} \supseteq \{EC_w \mid \langle w,j\rangle \in new\}$ and $i = p$ then
%           $Inf := \top$;
until $\varphi = \psi \vee \psi = \bot \vee Inf$.
```

Figure 8: The Procedure Check.

out from the queue *new* is put in the queue *old*. We will explain later the meaning of the two commented line in the procedure: they will ensure the termination.

The rules in $\exists$-reductions, applied by the procedure apply-$\exists$, build the solution adding new elements (variables) when it is necessary. Moreover, they establish to which subterms of $t(x, \vec{z})$ these elements (variables) have to belong.

The rule in $\forall$-reductions, applied by the procedure apply-$\forall$, ensures that all the new elements of x satisfy the universal conditions which are in $\psi_\forall$.

The rules in general-reductions, applied by the procedure apply-gen, determine whether there is a contradiction in the formula. Such a contradiction could derive from a wrong guess of the sets EC's and NC's or from the fact that the formula is not satisfiable. Intuitively, since all the possible choices are taken into consideration, if all the non-deterministic branches of Check(ψ) terminates with $\psi = \bot$, then the formula ψ is not satisfiable.

Lemma 5.2 *If all the non-deterministic branches of the* Check *procedure on input ψ terminate with $\psi = \bot$, then ψ is not satisfiable in any model of Ω.*

The third rule of $\exists$-reductions introduces a new variable, hence one, or more, of the non-deterministic branches of Check(ψ) could not terminate. This happens, for instance, when we are dealing with a formula ψ of $\mathcal{C}$ which has only solutions with an infinite number of elements.

From a set-theoretic point of view, recalling that we are not assuming Extensionality and that the Regularity axiom has not been replaced by any form of anti-foundation, we start by showing how to build a solution in the case that one of the possible non-deterministic branches of Check(ψ) terminates with $\psi \neq \bot$ or does not terminates. Let us use the term *infinitary formula* [3] to refer to a formula in our language in which we allow conjunctions of infinitely many literals. We extend the class $\mathcal{C}$ to the class $\mathcal{C}_\omega$ of generalized formulae in which we allow an infinite number of $\vec{v}$ variables. In particular $\psi^\omega(x, \vec{z}, \vec{v})$ is in $\mathcal{C}_\omega$ if and only if $\psi^\omega(x, \vec{z}, \vec{v})$ is a generalized formula of the form

$$\psi^\omega_\forall(x, \vec{v}) \wedge x \subseteq \wp(x) \wedge \psi^\omega_{\mathsf{term}}(x, \vec{z}, \vec{v}) \wedge \psi^\omega_{\mathsf{var}}(\vec{v}),$$

where $\psi^\omega_{\mathsf{var}}(\vec{v})$ and $\psi^\omega_{\mathsf{term}}(x, \vec{z}, \vec{v})$ can be conjunctions of ω literals, provided that there are only a finite number of variables in $\vec{z}$.

Consider the formula ψ^ω obtained after at most ω iterations of the repeat-loop of Check(ψ). Each model of Ω in which ψ^ω is satisfied is also a model of Ω in which ψ is satisfied, since ψ is a subformula of ψ^ω.

Let $G(\psi) = \langle N(\psi), E(\psi)\rangle$ be the following graph:

$$\begin{array}{rcl} N(\psi) & = & FV(\psi^\omega) \\ E(\psi) & = & \{\langle u_1, u_2\rangle \mid u_2 \in u_1 \text{ is an atom in } \psi^\omega\}. \end{array}$$

Let $G_\omega(\psi)$ be a model of Ω obtained from $G(\psi)$ as described in Section 3.

[3]The use of an infinitary language here is justified by convenience in dealing with quantifiers only, being easily eliminable from our argument.

Lemma 5.3 *All the literals that are in $\psi^\omega \setminus \psi_\forall$ are in one of the following forms:*
- $v = v$, with $v \in \mathcal{V}$;
- $v_1 \neq v_2$, with $\{v_1, v_2\} \subseteq \mathcal{V}$ and $v_1 \not\equiv v_2$;
- $v \in s$, with $v \in \mathcal{V}$;
- $v \notin s$, with $v \in \mathcal{V}$;
- $x \subseteq \wp(x)$.

The above result is nothing but a formal proof of the fact that the rules in Check maps formulae of the class $\mathcal{C}$ into formulae of the class $\mathcal{C}$ (formulae of the class $\mathcal{C}_\omega$, if we allow infinite branches).

Lemma 5.4 *$G_\omega(\psi)$ is a model of Ω and a model of ψ^ω.*

Theorem 5.5 *If there exists a non-deterministic branch of Check(ψ) which terminates with $\psi \neq \bot$ or which does not terminate, then ψ is satisfiable in a model of Ω.*

Corollary 5.6 *A formula ψ in the class $\mathcal{C}$ is satisfiable in a model of Ω if and only if not all the non-deterministic branches of Check(ψ) terminates with $\psi = \bot$.*

Notice that in the case Check terminates after a finite number of iterations with $\psi \neq \bot$, we have implicitly proved that ψ has a finite model. In this case the whole model $G_\omega(\psi)$ is infinite but the part of model (the graph) used to give a solution is finite, since in a finite number of iterations we can only add a finite number of new variables.

6 The Procedure TCheck: Correctness and Termination

In the case that $\psi_\forall$ is in the class $\forall^2$ with no positive literals of equality (we use $\mathcal{C}^2_{no=}$ to refer to this class) we modify the procedure Check in order to obtain a procedure TCheck which decides the satisfiability problem. This means that in this particular case we are able to add a termination condition which ensures the existence of a solution. The procedure TCheck is obtained from Check by simply removing the comment symbols % at the beginning of the last three lines of the repeat-loop.

Lemma 6.1 *Let ψ be a formula in $\mathcal{C}$ such that in $\psi_\forall$ there are no literals of the form $y_h = y_k, y_h = v_k, v_h = y_k$, and let ψ^ω be a formula obtained after at most ω iterations of Check(ψ). In ψ^ω there are no literals of the form $v = v$.*

Lemma 6.2 *Let ψ be a formula in $\mathcal{C}^2_{no=}$. The procedure TCheck on input ψ always terminates.*

Lemma 6.3 *If there exists a non-deterministic branches of TCheck(ψ) which terminates with $Inf = \bot$ and which maps the formula ψ into the formula ψ', then there exists a non-deterministic branch of Check(ψ) which terminates and maps the formula ψ into the formula ψ'.*

Lemma 6.4 *If there exists a non-deterministic branch of* TCheck(ψ) *which terminates with $Inf = \top$, then there exists a branch of* Check(ψ) *which does not terminate.*

Lemma 6.5 *If all the non-deterministic branches of* TCheck *map ψ into $\bot$, then all the non-deterministic branches of* Check *terminate and map ψ into $\bot$.*

Theorem 6.6 *Let ψ be a formula in the class $\mathcal{C}^2_{no=}$. ψ is satisfiable in a model of Ω in which ψ is satisfiable if and only if there exists a non-deterministic branch of* TCheck *at the end of which $\psi \neq \bot$. Moreover, if there exists a non-deterministic branch of* TCheck *at the end of which $Inf = \bot$ and $\psi \neq \bot$, then ψ is finitely satisfiable in a model of Ω.*

Notice that in order to force termination in this particular case we check to which subsets of x the new variables we are adding belong. When we are in a situation in which all the variables we have to process present a situation similar to variables already processed we are able to generate an infinite branch. This is similar to what we did in [17], where we analyzed what we called *pictures*. But in that case from the existence of an infinite branch we were not able to deduce the existence of a solution. That problem was mainly caused by the presence of the $\wp$ operator on the l.h.s. of inclusions. This never arises when the formulae we consider are translations of modal formulae.

We are working on a general termination condition which added to Check gives a decision procedure for the whole class $\mathcal{C}$. However, the objects required to express this condition are much more complicate than the EC's, which remain at the basis of the condition. In particular this makes the correctness proof, on which we are working, much more hard than the one we give here for the class $\mathcal{C}^2_{no=}$.

7 Modal Consequences

Definition 7.1 *Let Λ be a first-order complete modal logic with first-order correspondent δ. If δ is of the form $\exists^*\forall^2\alpha$ such that α is in conjunctive normal form and there are not positive literals of equality in α, then we say that Λ is an $\exists^*\forall^2$-*equality-free *modal logic.*

Corollary 7.2 *If Λ is an $\exists^*\forall^2$-equality-free modal logic, then Λ is decidable.*

Some $\exists^*\forall^2$-equality-free modal logics are K, T, B.

In the case that Λ is a first-order complete modal logic with first-order correspondent δ in the class $\exists^*\forall^*$ it is possible to use the procedure Check to test $\vdash_\Lambda \gamma$. In the way that the procedure is presented here it is possible that it does not terminate. However, what is interesting is that it allows to deal in a uniform way with a large class of modal logics: we do not have to define a set of tableau rules for each modal logic; there is only one rule, the rule $(3_\exists)$, which takes care of the introduction of new worlds; while the rule in $\forall$-reduction manages the *encoding* of the axioms of the modal logic. Some famous modal logics in this class are $K4$, $S4$, $S5$, $K4.3$, $S4.3$ (no branching to the right).

8 Conclusion and Future Developments

The results presented in this paper go towards establishing a strong link between decision problems in Set Theory and Modal Logic. The two main tools used are the $\Box$-as-$\mathcal{P}$ translation method and the tableau technique, intended as a general tool for coordinating a semantic and syntactic interplay in the analysis of a formula. The set-theoretic approach, on the one hand, guarantees an entirely uniform point of view on modal logics characterized via Hilbert's axioms[4]. Such a point of view, on the other hand, allows a tableau-like analysis that, abstracting from the specific modal logic under consideration concentrates on the model construction in completely general and intuitive (set-theoretic) terms. In this respect consider, for example, the fact that the only rule introducing new variables in our procedure is $(3_\exists)$.

The decidability result presented here is an example of the possible fruitfulness of the approach and we are convinced that such a result can be generalized to the entire class of modal logic with first-order correspondent of the form $\exists^*\forall^*$. Major stumbling block towards this achievement is the specification of the correct termination condition of the tableau procedure presented in Section 5.

Further studies relative to the possibilities given by the technique presented here for classes of modal logics differently specified are also planned.

References

[1] P. Aczel. *Non-well-founded sets.*, volume 14 of *Lecture Notes, Center for the Study of Language and Information.* Stanford, 1988.

[2] J. Barwise and L. Moss. Hypersets. *The Math. Intelligencer*, 13(4):31–41, 1991.

[3] J. Barwise and L. Moss. *Vicious Circles. On the Mathematics of non-well-founded phenomena.* CSLI, Stanford, 1996.

[4] J. F. A. K. van Benthem. *Modal Logic and Classical Logic.* Bibliopolis and Atlantics Heights, 1985.

[5] J. F. A. K. van Benthem, G. D'Agostino, A. Montanari, and A. Policriti. Modal deduction in second-order logic and set theory-I. *Journal of Logic and Computation*, 7(2):251–265, 1997.

[6] J. F. A. K. van Benthem, G. D'Agostino, A. Montanari, and A. Policriti. Modal deduction in second-order logic and set theory-II. *Studia Logica*, 60(3):387–420, 1998.

[7] D. Cantone. A fast saturation strategy for set-theoretic tableaux. In D. Galmiche, editor, *Proceedings of TABLEAUX'97*, pages 122–137. Springer-Verlag LNAI, vol. 1227, 1997.

[8] D. Cantone and Zarba C. A new fast saturation tableau-based decision procedure for an unquantified fragment of set theory. In *Proc. of the International Workshop in First-Order Theorem Proving, FTP'98*, 1998.

[4]It is possible to specify modal logics by adding modal formulae as axioms to K (*à la* Hilbert) or, semantically, through a description of their models. For non-complete modal logics these characterizations are not equivalent.

[9] D. Cantone and Zarba C. A tableau calculus for integrating first-order and elementary set theory reasoning. In *Proceedings of TABLEAUX'00*, 2000.

[10] D. Cantone, A. Ferro, and E. G. Omodeo. *Computable Set Theory. Vol. 1.* Oxford University Press, 1989. Int. Series of Monographs on Computer Science.

[11] G. D'Agostino, A. Montanari, and A. Policriti. Set theoretic decidability results for modal theorem proving. In *Proceedings of ICTCS-95*, 1996.

[12] G. D'Agostino, A. Montanari, and A. Policriti. A set-theoretic translation method for polymodal logics. *Journal of Automated Reasoning*, 15:317–337, 1996.

[13] A. Ferro, E. G. Omodeo, and J. T. Schwartz. Decision Procedures for Elementary Sublanguages of Set Theory I. Multilevel Syllogistic and Some Extensions. *Comm. Pure App. Math.*, 33:599–608, 1980.

[14] M. Forti and F. Honsell. Set theory with free construction principles. *Annali Scuola Normale Superiore di Pisa, Cl. Sc.*, IV(10):493–522, 1983.

[15] E. G. Omodeo and A. Policriti. Solvable set/hyperset contexts: I. Some decision procedures for the pure, finite case. *Comm. Pure App. Math.*, 48(9-10):1123–1155, 1995. Special Issue in honor of J.T. Schwartz.

[16] C. Piazza and A. Policriti. Deciding modal logics using tableaux and set theory. UDMI/RR 37/00, University of Udine, 2000.

[17] C. Piazza and A. Policriti. Towards tableau-based decision procedures for non-well-founded fragments of set theory. In *Proceedings of TABLEAUX'00*, 2000.