A Proof Procedure for Hybrid Logic with Binders, Transitivity and Relation Hierarchies

Marta Cialdea Mayer

Università di Roma Tre, Italy

Abstract. A tableau calculus constituting a decision procedure for hybrid logic with the converse modalities, the global ones and a restricted use of the binder has been defined in a previous paper. This work shows how to extend such a calculus to multi-modal logic equipped with two features largely used in description logics, i.e. transitivity and relation inclusion assertions. An implementation of the proof procedure is also briefly presented, along with the results of some preliminary experiments.

1 Introduction

This work considers multi-modal hybrid languages (see, for instance, [3]) that, beyond the standard modalities, nominals, the satisfaction operator and the binder, include the converse modalities ($\Diamond^-_R$ and $\Box^-_R$), the global ones (E and A) and a feature largely used in description logics, i.e. the possibility of declaring an accessibility relation to be transitive and/or included in another one. Basic hybrid logic (with nominals only, beyond the modal operators $\Diamond$ and $\Box$) will be denoted by HL, and basic multi-modal hybrid logic by HL_m. Logics extending HL or HL_m with operators $O_1, \ldots, O_n$ (and their duals) are denoted by $\mathsf{HL}(O_1, \ldots, O_n)$ and $\mathsf{HL}_m(O_1, \ldots, O_n)$, respectively. Multi-modal languages including transitivity assertions and/or relation hierarchies are denoted in the same way, just including Trans (for transitivity) and/or $\sqsubseteq$ (for relation inclusion) among $O_1, \ldots, O_n$.

The satisfiability problem for formulae of any hybrid logic $\mathsf{HL}(O_1, \ldots, O_n)$ or $\mathsf{HL}_m(O_1, \ldots, O_n)$ – where $O_i \in \{@, \Diamond^-, \mathsf{E}\}$ is decidable [3]. Unfortunately, due to the high expressive power of the binder, $\mathsf{HL}(\downarrow)$ is undecidable [1, 4].

There are both semantic and syntactic restrictions allowing for regaining decidability of hybrid logic with the binder. Restricting the frame class is a way of restoring decidability, but the interplay with multi-modalities (or the addition of other operators) is not always harmless. For instance, $\mathsf{HL}(\downarrow)$ over transitive frames is decidable [18], but $\mathsf{HL}(@, \downarrow)$ and $\mathsf{HL}_m(\downarrow)$ are not [18, 17].

In [20] it is proved that the satisfiability problem for formulae in $\mathsf{HL}(@, \downarrow, \mathsf{E}, \Diamond^-)$ is decidable, provided that their negation normal form contains no universal operator (i.e. either $\Box$ or $\Box^-$ or A) scoping over a binder, that in turn has scope over a universal operator. Such a fragment of hybrid logic is denoted by $\mathsf{HL}(@, \downarrow, \mathsf{E}, \Diamond^-) \setminus \Box\downarrow\Box$. The result is proved by showing that there exists a satisfiability preserving translation of $\mathsf{HL}(@, \downarrow, \mathsf{E}, \Diamond^-) \setminus \Box\downarrow\Box$ into $\mathsf{HL}(@, \downarrow, \mathsf{E}, \Diamond^-) \setminus \downarrow\Box$, i.e.

M.P. Bonacina (Ed.): CADE 2013, LNAI 7898, pp. 76–90, 2013.

the set of formulae in negation normal form where no universal operator occurs in the scope of a binder. The standard translation of hybrid logic into first order classical logic [1, 20] maps, in turn, formulae in $\mathsf{HL}(@, \downarrow, \mathsf{E}, \Diamond^-) \setminus \downarrow\Box$ into universally guarded formulae, that have a decidable satisfiability problem [12].

Decidability of $\mathsf{HL}_m(@, \downarrow, \mathsf{E}, \Diamond^-) \setminus \Box\downarrow\Box$ can be proved by the same reasoning, and the separate addition of either relation hierarchies or transitive relations can easily be shown to stay decidable, by reduction to the first order guarded fragment and by resorting to results already proved in the literature [19]. However, such results do not directly allow for concluding whether the logic including both features is still decidable.

This work is a continuation of previous works, where terminating tableau calculi for decidable fragments of Hybrid Logic with the binder have been defined [8, 9]. In particular, [9] presents a tableau calculus constituting a satisfiability decision procedure for $\mathsf{HL}(@, \downarrow, \mathsf{E}, \Diamond^-) \setminus \Box\downarrow\Box$. Such a procedure is here extended to multi-modal hybrid logic $\mathsf{HL}_m(@, \downarrow, \mathsf{E}, \Diamond^-, \mathsf{Trans}, \sqsubseteq) \setminus \Box\downarrow\Box$: a tableau calculus is presented, which terminates and is sound and complete for formulae in the fragment $\mathsf{HL}_m(@, \downarrow, \mathsf{E}, \Diamond^-, \mathsf{Trans}, \sqsubseteq) \setminus \downarrow\Box$, i.e. formulae in negation normal form where no occurrence of a universal operator is in the scope of a binder, with the addition of transitivity assertions and relation hierarchies. A preprocessing step along the lines of [20] turns the calculus into a satisfiability decision procedure for the fragment $\mathsf{HL}_m(@, \downarrow, \mathsf{E}, \Diamond^-, \mathsf{Trans}, \sqsubseteq) \setminus \Box\downarrow\Box$. Soundness, completeness and termination of the tableaux calculus thus imply that the satisfiability problem for the fragment of multi-modal hybrid logic $\mathsf{HL}_m(@, \downarrow, \mathsf{E}, \Diamond^-, \mathsf{Trans}, \sqsubseteq) \setminus \Box\downarrow\Box$ is decidable. The proof procedure has been implemented in a prover called Sibyl, which will be briefly presented along with the results of some preliminary experiments.

The language of $\mathsf{HL}_m(@, \downarrow, \mathsf{E}, \Diamond^-, \mathsf{Trans}, \sqsubseteq) \setminus \Box\downarrow\Box$ subsumes the description logic $\mathcal{SHOI}$ enriched with restricted occurrences of the binder, and allows for representing some interesting frame properties, such as, for instance, symmetry ($R^- \sqsubseteq R$), reflexivity ($\mathsf{A}\downarrow x.\Diamond_R x$), "at most" restrictions on the number of states ($\mathsf{E}\downarrow x_1. \ldots \mathsf{E}\downarrow x_n.\mathsf{A}(x_1 \vee \cdots \vee x_n)$), and "at least" restrictions on the number of R-successors of each state ($\mathsf{A}\downarrow x.\Diamond_R\downarrow y_1.(x : \Diamond_R(\neg y_1 \wedge \downarrow y_2.(x : \Diamond_R(\neg y_1 \wedge \neg y_2 \wedge \downarrow y_3. \ldots)))))$.

This section concludes with a brief introduction to the syntax and semantics of multi-modal hybrid logic with transitive relations and inclusion assertion. Well-formed expressions of $\mathsf{HL}_m(@, \downarrow, \mathsf{E}, \Diamond^-, \mathsf{Trans}, \sqsubseteq)$ are partitioned into two categories: *formulae* (for which the metasymbols F, G are used) and *assertions*.

Formulae are built out of a set PROP of propositional letters, a set NOM of nominals, an infinite set VAR of state variables, and a set REL of relation symbols (all such sets being mutually disjoint), and defined by the following grammar:

$$\begin{aligned} F := {} & p \mid a \mid x \mid \neg F \mid F \wedge F \mid F \vee F \mid \Diamond_R F \mid \Box_R F \\ & \mid \Diamond_R^- F \mid \Box_R^- F \mid \mathsf{E}F \mid \mathsf{A}F \mid a{:}\, F \mid x{:}\, F \mid \downarrow x.F \end{aligned}$$

where $p \in \mathsf{PROP}$, $a \in \mathsf{NOM}$, $x \in \mathsf{VAR}$ and $R \in \mathsf{REL}$. In this work, the notation $t{:}\, F$ is used (for $t \in \mathsf{NOM} \cup \mathsf{VAR}$) rather than $@_t F$. We use metavariables a, b, c for nominals, x, y, z for state variables and R, S, P for relation symbols.

If F is a formula, x a state variable and a a nominal, then $F[a/x]$ denotes the formula obtained from F by substituting a for every free occurrence of x (an occurrence of x is free if it is not in the scope of a $\downarrow x$). If $a_0, \ldots, a_n, b_0, \ldots, b_n$ are nominals, then $F[b_0/a_0, \ldots, b_n/a_n]$ denotes the formula obtained from F by simultaneously replacing b_i for every occurrence of a_i.

Assertions are either *transitivity assertions,* of the form $\mathsf{Trans}(R)$, for $R \in \mathsf{REL}$, or *inclusion assertions,* of either form $R \sqsubseteq S$ or $R^- \sqsubseteq S$, for $R, S \in \mathsf{REL}$. Here, R^- is intended to denote the inverse of the relation denoted by R, i.e. the set of pairs of states $\langle w, w' \rangle$ such that $\langle w', w \rangle$ is in the relation denoted by R. Note that inverse relations are allowed only on the left of the $\sqsubseteq$ symbol. This is only a syntactical restriction, since $R^- \sqsubseteq S^-$ is equivalent to $R \sqsubseteq S$, and $R \sqsubseteq S^-$ is equivalent to $R^- \sqsubseteq S$.

An *interpretation* $\mathcal{M}$ of an $\mathsf{HL}_m(@, \downarrow, \mathsf{E}, \Diamond^-, \mathsf{Trans}, \sqsubseteq)$ language is a tuple $\langle W, \rho, N, I \rangle$ where W is a non-empty set (whose elements are the *states* of the interpretation), ρ is a function mapping every $R \in \mathsf{REL}$ to a binary relation on W ($\rho(R) \subseteq W \times W$), N is a function $\mathsf{NOM} \to W$ and I a function $W \to 2^{\mathsf{PROP}}$.

If $\mathcal{M} = \langle W, \rho, N, I \rangle$ is an interpretation, $w \in W$, σ is a variable assignment for $\mathcal{M}$ (i.e. a function $\mathsf{VAR} \to W$) and F is a formula, the relation $\mathcal{M}_w, \sigma \models F$ is defined adding the following clauses to the usual definition of the classical operators:

1. $\mathcal{M}_w, \sigma \models p$ if $p \in I(w)$, for $p \in \mathsf{PROP}$.
2. $\mathcal{M}_w, \sigma \models a$ if $N(a) = w$, for $a \in \mathsf{NOM}$.
3. $\mathcal{M}_w, \sigma \models x$ if $\sigma(x) = w$, for $x \in \mathsf{VAR}$.
4. $\mathcal{M}_w, \sigma \models a{:}\, F$ if $\mathcal{M}_{N(a)}, \sigma \models F$, for $a \in \mathsf{NOM}$.
5. $\mathcal{M}_w, \sigma \models x{:}\, F$ if $\mathcal{M}_{\sigma(x)}, \sigma \models F$, for $x \in \mathsf{VAR}$.
6. $\mathcal{M}_w, \sigma \models \downarrow x.F$ if $\mathcal{M}_w, \sigma^w_x \models F$, where σ^w_x is the variable assignment such that $\sigma^w_x(x) = w$ and, for $y \neq x$, $\sigma^w_x(y) = \sigma(y)$.
7. $\mathcal{M}_w, \sigma \models \Box_R F$ if for every w' such that $\langle w, w' \rangle \in \rho(R)$, $\mathcal{M}_{w'}, \sigma \models F$.
8. $\mathcal{M}_w, \sigma \models \Diamond_R F$ if there exists w' such that $\langle w, w' \rangle \in \rho(R)$ and $\mathcal{M}_{w'}, \sigma \models F$.
9. $\mathcal{M}_w, \sigma \models \Box_{\overline{R}} F$ if for every w' such that $\langle w', w \rangle \in \rho(R)$, $\mathcal{M}_{w'}, \sigma \models F$.
10. $\mathcal{M}_w, \sigma \models \Diamond_{\overline{R}} F$ if there exists w' such that $\langle w', w \rangle \in \rho(R)$ and $\mathcal{M}_{w'}, \sigma \models F$.
11. $\mathcal{M}_w, \sigma \models \mathsf{A}F$ if $\mathcal{M}_{w'}, \sigma \models F$ for all $w' \in W$.
12. $\mathcal{M}_w, \sigma \models \mathsf{E}F$ if $\mathcal{M}_{w'}, \sigma \models F$ for some $w' \in W$.

Two formulae F and G are logically equivalent when, for every interpretation $\mathcal{M}$, assignment σ and state w of $\mathcal{M}$: $\mathcal{M}_w, \sigma \models F$ if and only if $\mathcal{M}_w, \sigma \models G$. Every formula in $\mathsf{HL}_m(@, \downarrow, \mathsf{E}, \Diamond^-)$ is logically equivalent to a formula in negation normal form (NNF), where negation appears only in front of atoms. Therefore, considering only formulae in NNF does not restrict the expressive power of the language.

If $\mathcal{A}$ is a set of assertions, an interpretation $\langle W, \rho, N, I \rangle$ is a model of $\mathcal{A}$ if:

1. for all $R \in \mathsf{REL}$ such that $\mathsf{Trans}(R) \in \mathcal{A}$, $\rho(R)$ is a transitive relation;
2. for all $R, S \in \mathsf{REL}$, if $R \sqsubseteq S \in \mathcal{A}$, then $\rho(R) \subseteq \rho(S)$;
3. for all $R, S \in \mathsf{REL}$ and all $w, w' \in W$, if $R^- \sqsubseteq S \in \mathcal{A}$ and $\langle w, w' \rangle \in \rho(R)$, then $\langle w', w \rangle \in \rho(S)$.

Finally, if F is a formula and $\mathcal{A}$ a set of assertions, $\{F\} \cup \mathcal{A}$ is satisfiable if there exist a model $\mathcal{M}$ of $\mathcal{A}$ and a state w of $\mathcal{M}$ such that $\mathcal{M}_w \models F$ (i.e. $\mathcal{M}_w, \sigma \models F$ for every variable assignment σ).

2 The Tableau Calculus

This section shows how to extend the system described in [9] to the presence of transitivity and inclusion assertions. The expansion rules that will be introduced to treat assertions are similar to the analogous rules presented by [13–16]. However, their addition to a terminating calculus dealing also with syntactically restricted occurrences of the binder is a novelty.

The presentation will be as self contained as possible, therefore it overlaps with the description given in [9] in many points. However, since some of the basic notions underlying the calculus are quite involved, they are not given a completely formal account.

A tableau is a set of branches, and a *tableau branch* is a sequence of *nodes* $n_0, n_1, \ldots$, where each node is labelled either by an assertion or a ground *satisfaction statement*, i.e. a formula of the form $a{:}\, F$, where no state variable occurs free in F. The nominal a in a satisfaction statement $a{:}\, F$ is called the *outermost nominal* of the formula. Node labels are always formulae in NNF. The reason why a branch is not simply a set of formulae will be briefly explained in the sequel.

If n occurs before m in a branch, we write $n < m$. The label of the node n is denoted by label(n). The notation $(n)\, a{:}\, F$ is used to denote the node n, and simultaneously say that its label is $a{:}\, F$. If a node $(n)\, a : F$ is in a branch, then the nominal a is said to label the formula F in the branch.

In order to give a more compact presentation of the expansion rules, some notions and abbreviations will be adopted. Relation symbols will also be called *forward relations* (and have *positive sign*) and the inverse of relation symbols *backward relations* (with *negative sign*). A *relation* is either a forward or backward relation. Relations are denoted by boldface letters: $\mathbf{R}$ is a meta-symbol used to denote either R itself or its inverse R^-. The following table defines some shorthands for formulae and assertions that will be used in the sequel.

$$a \Rightarrow_{\mathbf{R}} b \equiv_{def} \begin{cases} a{:}\, \Diamond_R b \text{ if } \mathbf{R} = R \\ b{:}\, \Diamond_R a \text{ if } \mathbf{R} = R^- \end{cases} \qquad a{:}\, \Diamond_{\mathbf{R}} F \equiv_{def} \begin{cases} a{:}\, \Diamond_R F \text{ if } \mathbf{R} = R \\ a{:}\, \Diamond_{\overline{R}} F \text{ if } \mathbf{R} = R^- \end{cases}$$

$$a{:}\, \Box_{\mathbf{R}} F \equiv_{def} \begin{cases} a{:}\, \Box_R F \text{ if } \mathbf{R} = R \\ a{:}\, \Box_{\overline{R}} F \text{ if } \mathbf{R} = R^- \end{cases} \qquad \mathbf{R} \sqsubseteq \mathbf{S} \equiv_{def} \begin{cases} R \sqsubseteq S & \text{if } \mathbf{R} \text{ and } \mathbf{S} \text{ have the same sign} \\ R^- \sqsubseteq S & \text{if } \mathbf{R} \text{ and } \mathbf{S} \text{ have different signs} \end{cases}$$

Let F be a ground hybrid formula in NNF and $\mathcal{A}$ a set of assertions. A tableau for $\{F\} \cup \mathcal{A}$ is initialized with a single branch, constituted by the node $(n_0)\, a_0 \colon F$, where a_0 is a new nominal, followed by nodes labelled by the assertions in $\mathcal{A}$ and then expanded according to the following *Assertion rules*:

$$\frac{}{R \sqsubseteq R}\ \mathsf{Rel}_0 \qquad \frac{\mathbf{R} \sqsubseteq \mathbf{S} \quad \mathbf{S} \sqsubseteq \mathbf{P}}{\mathbf{R} \sqsubseteq \mathbf{P}}\ \mathsf{Rel}$$

(note that Rel actually stands for four rules, according to the relation signs). Such rules complete the inclusion assertions in $\mathcal{A}$ by the reflexive and transitive closure of $\sqsubseteq$. The formula $a_0 \colon F$ is the *initial formula* of the tableau.

A tableau is expanded by application of the rules in Tables 1 and 2, which are applied to a given branch.

Table 1. Expansion rules: first group

$$\frac{(n)\, a \colon (F \wedge G)}{\begin{array}{l}(m_0)\, a \colon F \\ (m_1)\, a \colon G\end{array}}\ (\wedge) \qquad \frac{(n)\, a \colon (F \vee G)}{(m_0)\, a \colon F \quad \mid \quad (m_1)\, a \colon G}\ (\vee)$$

$$\frac{(n)\, a \colon b \colon F}{(m)\, b \colon F}\ (@) \qquad \frac{(n)\, a \colon \downarrow x.F}{(m)\, a \colon F[a/x]}\ (\downarrow)$$

$$\frac{(n)\, a \colon \Box_{\mathbf{R}} F \quad (m)\, a \Rightarrow_{\mathbf{R}} b}{(k)\, b \colon F}\ (\Box)$$

$$\frac{(n)\, a \colon \Diamond_R F}{\begin{array}{l}(m_0)\, a \colon \Diamond_R b \\ (m_1)\, b \colon F\end{array}}\ (\Diamond) \qquad \frac{(n)\, a \colon \Diamond_R^- F}{\begin{array}{l}(m_0)\, b \colon \Diamond_R a \\ (m_1)\, b \colon F\end{array}}\ (\Diamond^-)$$

where b is a fresh nominal (not applicable if F is a nominal) — where b is a fresh nominal

$$\frac{(n)\, a \colon \mathsf{A}F}{(m)\, b \colon F}\ (\mathsf{A}) \qquad \frac{(n)\, a \colon \mathsf{E}F}{(m)\, b \colon F}\ (\mathsf{E})$$

where b occurs in the branch — where b is a fresh nominal

$$\frac{\begin{array}{c}[\mathcal{B}] \\ (n)\, a \colon b\end{array}}{\mathcal{B}[b/a]}\ (=)$$

Most rules are standard, and their reading is standard too. Note that when the formulation of a rule contains (boldface) relations, it actually stands for different rules, according to the relations signs. The rules of Table 1 are the same as those presented in [9], but for the fact that the modal rules ($\Box$, $\Diamond$ and $\Diamond^-$) are here

reformulated to address the multi-modal case. The *equality rule* (=) does not add any node to the branch, but modifies the labels of its nodes. The schematic formulation of this rule in Table 1 indicates that it can be fired whenever a branch $\mathcal{B}$ contains a *nominal equality* of the form $a{:}\, b$ (with $a \neq b$); as a result of the application of the rule, every node label F in $\mathcal{B}$ is replaced by $F[b/a]$.

Formulae of the form $\Box_{\mathbf{R}} F$ and $\mathsf{A}F$ are called *universal formulae*; nodes whose labels have the form $a{:}\, G$, where G is a universal formula, are *universal nodes* and the rules $\Box$ and A are called *universal rules*. When the A rule is applied producing a node labelled by a formula of the form $b : F$, it is said to *focus* on b (and b is the focused nominal of the inference). The $\Diamond$, $\Diamond^-$ and E rules are called *blockable rules*, formulae of the form $a{:}\, \Diamond_R F$, where F is not a nominal, $a{:}\, \Diamond_{\overline{R}} F$, and $a{:}\, \mathsf{E}F$ are *blockable formulae* and a node labelled by a blockable formula is a *blockable node*. A formula of the form $a{:}\, \Diamond_R b$, where R is a forward relation, is called a *relational formula*.

The Trans rule of Table 2 deals with transitive relations and can be seen as a reformulation (in the presence of inclusion assertions) of the $\Box$ rule for transitive modal logics (a particular case of this rule is when $R = S$). In the Link rule, that deals with inclusion assertions, R is always a forward relation.

Table 2. Expansion rules: second group

$$\frac{(n)\, a{:}\, \Diamond_R b \quad (i)\, R \sqsubseteq \mathbf{S}}{(m)\, a \Rightarrow_{\mathbf{S}} b}\ (\mathsf{Link})$$

$$\frac{(n)\, a{:}\, \Box_{\mathbf{S}} F \quad (m)\, a \Rightarrow_{\mathbf{R}} b \quad (t)\, \mathsf{Trans}(R) \quad (i)\, \mathbf{R} \sqsubseteq \mathbf{S}}{(k)\, b{:}\, \Box_{\mathbf{R}} F}\ (\mathsf{Trans})$$

The premiss n of either the $\Box$ or Trans rules is called the *major premiss*, and m the *minor premiss* of the rule. In an application of the Link rule, n is the *logical premiss*. The premisses i and t, in the rules of Table 2, are the *side premisses* of the rules.

The formulation of the Trans rule is very close to the corresponding one used in description logics, where in fact "roles" include both *role names* (corresponding to relation symbols) and the inverse of role names, and inverse roles may also occur in role inclusion axioms. The abbreviation $a \Rightarrow_{\mathbf{R}} b$, however, does not have exactly the same meaning as the corresponding premiss used in the rule treating transitivity in description logics [13, 14] (a similar approach is adopted in [15]), consisting of the meta-notion "b is an $\mathbf{R}$-neighbour of a". There are two main differences between the two approaches. First of all, the semantical notion of accessibility between two states is here given a "canonical representation" in the object language (a choice already made in [8, 9]): the fact that a state a is R-related to b is represented by the *relational formula* $a{:}\, \Diamond_R b$. Though semantically equivalent to $b{:}\, \Diamond_{\overline{R}} a$, the latter is not a relational formula, i.e. it is not the

canonical representation of an R-relation. This is reflected by the fact that the $\Diamond$ rule cannot be applied to a relational formula, while $b\colon \Diamond_R^- a$ can be expanded by means of the $\Diamond^-$ rule. Moreover, in the present work, the notation $a \Rightarrow_{\mathbf{R}} b$ is only an abbreviation for a relational formula, which does not take subrelations into account: it may be the case that $a \Rightarrow_{\mathbf{S}} b$ belongs to a given branch $\mathcal{B}$ for some $\mathbf{S} \sqsubseteq \mathbf{R}$, and yet $a \Rightarrow_{\mathbf{R}} b$ does not. The fact that, in the present work, no meta-notion is used to represent "$\mathbf{R}$-neighbours" is responsible for the presence of the Link rules, that have no counterpart in [13–15].

The first node of a branch $\mathcal{B}$ is called the *top node* and its label the *top formula* of $\mathcal{B}$. Nominals occurring in the top formula are called *top nominals*. The notion of top nominal is relative to a tableau branch, because applications of the equality rule may change the top formula, hence the set of top nominals.

A branch is *closed* whenever it contains, for some nominal a, either a pair of nodes $(n)\, a\colon p$, $(m)\, a\colon \neg p$ for some $p \in \mathsf{PROP}$, or a node $(n)\, a\colon \neg a$. As usual, it is assumed that a closed branch is never expanded further. A branch which is not closed is *open*. A branch is *complete* when it cannot be further expanded.

Provided that the initial formula is in $\mathsf{HL}_m(@, \downarrow, \mathsf{E}, \Diamond^-) \setminus \downarrow\Box$, the calculus enjoys the following important *strong subformula property*, used to prove both termination and completeness: every universal formula occurring in a tableau branch is obtained from a subformula of the top formula F_0 of the branch by possibly replacing operators $\Box_{\mathbf{R}}$ with $\Box_{\mathbf{S}}$, for some relation $\mathbf{S}$ in the language of F_0. Treating nominal equalities by means of substitution, like in [6, 7, 9, 11], is essential to ensure such a property. By the effect of substitution, however, distinct node labels may become equal, though the corresponding nodes are still distinct elements of the branch.

The reason why nodes with the same label do not collapse is that they must be arrangeable in a tree-like structure, where each node has at most one parent. The relation on nodes inducing such a structure (see Definition 2) is used to define indirect blocking (Definition 3). Termination is in fact achieved by means of a form of anywhere blocking with indirect blocking.

Direct blocking is a relation between nodes in a tableau branch, holding whenever the respective labels (formulae) are equal up to (a proper form of) nominal renaming. Essentially, in order for a node $(n)\, F$ to (directly) block $(m)\, G$ in a branch $\mathcal{B}$, it must be the case that $G = F[a_1/b_1, \ldots, a_n/b_n]$, where $a_1, \ldots, a_n$, $b_1 \ldots, b_n$ are *non-top* nominals such that, for all $i = 1, \ldots, n$, a_i and b_i label the same set of propositions in PROP and the same formulae of the form $\Box_{\mathbf{R}} F$. More precisely:

Definition 1 (Nominal compatibility and mappings). *If $\mathcal{B}$ is a tableau branch, then:*

1. *two nominals a and b are* compatible *in $\mathcal{B}$ if they label the same propositions in* PROP *and the same formulae of the form $\Box_{\mathbf{R}} F$.*
2. *A mapping π for $\mathcal{B}$ is an* injective *function from* non-top *nominals to* non-top *nominals such that for all a, a and $\pi(a)$ are compatible in $\mathcal{B}$. Mappings*

are extended to act on formulae in the obvious way: $\pi(F)$ is the formula obtained by substituting $\pi(a)$ for a in F, for every nominal a.

3. *A mapping π for $\mathcal{B}$ maps a formula F to a formula G if $\pi(F) = G$ and π is the identity for all nominals which do not occur in F.*
4. *A formula F can be mapped to a formula G in $\mathcal{B}$ if there exists a mapping π for $\mathcal{B}$ mapping F to G.*

The (direct) blocking restriction forbids the application of a blockable rule to a node n, whenever the label of a node $m < n$ can be mapped to label(n).

As already mentioned before, indirect blocking relies on a partial order on the nodes of a branch $\mathcal{B}$, called the *offspring relation* and denoted by $\prec_{\mathcal{B}}$, which arranges them into a family of trees, where non-terminal nodes are blockable nodes. Every tree is rooted at a node called a *root node* (a node with no *parents* w.r.t. the offspring relation). When a blockable rule is applied, the generated nodes are *children* of the expanded node. All the other rules generate *siblings* of one of the premisses of the inference (two nodes are siblings either if they are both root nodes or they have the same parent).

Properly, the offspring relation and blockings are defined by a mutual recursion on branch construction: if $\mathcal{B}'$ is a branch obtained by expanding $\mathcal{B}$, the definition of $\prec_{\mathcal{B}'}$ assumes that the set of blocked nodes in $\mathcal{B}$ is already defined, and indirectly blocked nodes in $\mathcal{B}$ depend on the relation $\prec_{\mathcal{B}}$. This is due to the presence of the A rule, for which a *minor premiss* must be defined, since nodes added to a branch $\mathcal{B}$ by an application $\mathcal{I}$ of the A rule are siblings of such a minor premiss (in the new branch $\mathcal{B}'$ obtained from the expansion); but, in order to determine the minor premiss of $\mathcal{I}$ it is necessary to know which nodes are blocked in $\mathcal{B}$.

The presentation that follows is somewhat simplified, and the reader is referred to [9] for the more formal approach. Let us assume that when the A rule is applied, beyond the premiss shown in Table 1, the branch contains a node called the minor premiss of the rule application (which will be defined further on, in Definition 5).

Definition 2 (Offspring relation). *Let $\mathcal{B}$ be a tableau branch.*

1. *Every node already contained in the initial branch from which $\mathcal{B}$ is obtained (i.e. its top node and all the nodes labelled by assertions) is a root node.*
2. *If a node n has been added to $\mathcal{B}$ by application of a blockable rule to node m, then $m \prec_{\mathcal{B}} n$ (n is a child of m and m is the parent of n).*
3. *If n has been added to $\mathcal{B}$ by application of either a universal rule or the* Trans *rule, whose minor premiss is m, then n is a sibling of m (i.e., if m is a root node, then n is a root node too; otherwise, if $k \prec_{\mathcal{B}} m$, then $k \prec_{\mathcal{B}} n$).*
4. *If n has been added to $\mathcal{B}$ by application of any other rule of table 1 (i.e. any other single-premiss rule) to node m, then n is a sibling of m.*
5. *If n has been added to $\mathcal{B}$ by application of the* Link *rule, then n is a sibling of the logical premiss of the inference.*

It is worth pointing out that an application of either the Trans rule or a universal one produces a sibling of the minor premiss of the inference, and not the

major one. This is an essential feature of the offspring relation, needed to prove termination.

The notions of direct and indirect blocking can now be defined.

Definition 3 (Direct and indirect blocking). *Let $\mathcal{B}$ be a tableau branch. The set of directly and indirectly blocked nodes in $\mathcal{B}$ is defined by induction on the (total) order $<$ on the nodes of $\mathcal{B}$:*

- *n is blocked if it is either directly or indirectly blocked.*
- *n is directly blocked by m if n is a blockable node, $m < n$, m is not blocked and* label(m) *can be mapped to* label(n) *in $\mathcal{B}$; n is directly blocked in $\mathcal{B}$ if it is directly blocked by some m in $\mathcal{B}$.*
- *n is indirectly blocked if it is not directly blocked and it has an ancestor w.r.t. $\prec_{\mathcal{B}}$ which is blocked.*

An indirectly blocked node is called a phantom node *(or, simply, a phantom).*

It is worth noticing that a node is a phantom if and only if all its siblings are phantoms too.

The application of the expansion rules is restricted by the conditions defined next. Restrictions **R1**–**R4** are essentially the same as those formulated in [9]. The restrictions concerning the new rules are formulated apart (**R5**–**R6**).

Definition 4 (Restrictions on the expansion rules). *The expansion of a tableau branch $\mathcal{B}$ is subject to the following restrictions:*

R1. *no node labelled by a formula already occurring in $\mathcal{B}$ as the label of a* non-phantom *node is ever added to $\mathcal{B}$.*
R2. *Blockable nodes can be expanded at most once in a branch.*
R3. *A phantom node cannot be expanded by means of a single-premiss rule (including the equality rule), nor can it be used as the minor premiss of a universal rule.*
R4. *A blockable node n cannot be expanded if it is directly blocked in $\mathcal{B}$.*
R5. *A phantom node cannot be used as the minor premiss of the* Trans *rule.*
R6. *A phantom node cannot be used as the logical premiss of the* Link *rule.*

Finally, we only need to define the minor premiss of an application of the A rule.

Definition 5. *If $\mathcal{B}$ is obtained from $\mathcal{B}'$ by means of an application $\mathcal{I}$ of the A rule focusing on the nominal b, then the minor premiss of $\mathcal{I}$ is the first non-phantom node in $\mathcal{B}'$ where b occurs.*

Note that, as a particular case of restriction **R3**, the A rule cannot focus on a nominal which only occurs in phantom nodes in the branch. Consequently, thanks to restriction **R3**, every application of the A rule has a minor premiss.

Due to space restrictions, the termination and completeness proofs cannot be included in this work, but can be found in [10]. Here, only a short proof sketch is included.

Theorem 1 (Termination). *If the initial formula of a tableau is in the fragment* $\mathsf{HL}(@, \downarrow, \mathsf{E}, \Diamond^-) \setminus \downarrow\Box$*, then every tableau branch has a bounded depth and tableau construction always terminates.*

Termination if proved by showing that the nodes of a branch $\mathcal{B}$ are arranged by the offspring relation into a bounded sized set of trees, each of which has bounded width and bounded depth. This holds because a branch is not a set of formulae, but nodes, and each node has at most one parent. If nodes labelled by the same formula collapsed into a single branch element, such an element might have multiple parents.[1]

The drawback is that the reasoning proving that any node has a bounded number of siblings is not as simple as it would be if dealing with sets of formulae. It relies in an essential way on the fact that universal rules do not generate siblings of their major premisses and, thanks to the mentioned strong subformula property, the number of universal formulae occurring in a tableau branch is bounded.

In order to prove that tree depth is also bounded, it is shown that the size of any set of blockable nodes which may occur in a tableau branch, and such that none of its elements blocks another one, is bounded. This holds for two reasons. First of all, the calculus enjoys a *weak subformula property*: for any non-relational formula $a\colon F$ occurring in a tableau branch, F is obtained from a subformula of the top formula F_0 of the branch by replacing free variables with nominals and, possibly, operators $\Box_{\mathbf{R}}$ with $\Box_{\mathbf{S}}$, for some relation $\mathbf{S}$ in the language of F_0. Secondly, the strong subformula property ensures that the number of nominal compatibility classes is bounded.

Theorem 2 (Completeness). *Let F be a formula and $\mathcal{A}$ a set of assertions. If $\{F\} \cup \mathcal{A}$ is in $\mathsf{HL}_m(@, \downarrow, \mathsf{E}, \Diamond^-, \mathsf{Trans}, \sqsubseteq) \setminus \downarrow\Box$ and is unsatisfiable, then any complete tableau for $\{F\} \cup \mathcal{A}$ is closed.*

In order to prove that the calculus is complete, it is shown – like in [9] – how to extend a subset $\mathcal{N}_0$ of any complete and open branch $\mathcal{B}$ in such a way that every directly blocked node is added a suitable "witness" (the witness(es) of a blockable node n can be viewed simply as node(s) which could by obtained by application of the corresponding blockable rule to n). The fact that the labels of blocked and blocking nodes are not necessarily identical does not allow taking the witness of the blocking node as a witness of the blocked one. Nor can a model be simply built from a set of states consisting of equivalence classes of nominals, where two nominals are in the same class whenever some blocking mapping maps one to the other: two nominals a and b may be compatible even if the branch contains a node labelled by $a\colon \neg b$.

The initial set of the construction, $\mathcal{N}_0$, is the union of the non-phantom nodes in $\mathcal{B}$ and the nodes of the form $(n)\, a\colon F$, with a occurring in some non-phantom node in $\mathcal{B}$ and either F has the form $\Box_{\mathbf{R}} G$ or $F \in \mathsf{PROP}$. $\mathcal{N}_0$ is extended by steps,

[1] For a similar reason it is not possible to block nominals instead of nodes: two nominals with different parents may become equal by substitution.

constructing a (possibly infinite) sequence of sets of nodes $\mathcal{N}_0 \subseteq \mathcal{N}_1 \subseteq \mathcal{N}_2 \ldots$, where each $\mathcal{N}_{i+1}$ is obtained from $\mathcal{N}_i$ by (fairly) choosing a blockable node n with no witness in $\mathcal{N}_i$. The construction ensures that there exists a node $n_0 \in \mathcal{N}_0$ whose label can be mapped to label(n) in $\mathcal{N}_i$. The blocking mapping is then used to add new nodes and obtain $\mathcal{N}_{i+1}$, in such a way that n has a witness in $\mathcal{N}_{i+1}$. It is finally shown how to build a model of the initial formula from the union of the sets $\mathcal{N}_i$ (due to the presence of assertions, the construction is quite different from the corresponding one in [9]).

We conclude with some examples illustrating the calculus in action. The first simple one below shows the interplay between the Trans and Link rules. It consists of the closed one-branch tableau represented below for the formula $\Diamond_S \Diamond_S p \wedge \Box_S \neg p$, together with the assertions $\mathsf{Trans}(R)$, $R \sqsubseteq S$, $S \sqsubseteq R$. The notations $n \rightsquigarrow^{\mathcal{R}} m$ or $(n_1, \ldots, n_k) \rightsquigarrow^{\mathcal{R}} m$, used in the rightmost column below, means that the addition of node m is due to the application of rule $\mathcal{R}$ to node n (or nodes $n_1, \ldots, n_k$). Nodes 0–4 constitute the initial tableau. The branch is closed because of nodes 11 and 15.

(0) $a\colon (\Diamond_S \Diamond_S p \wedge \Box_S \neg p)$		(8) $a\colon \Diamond_S b$	$6 \rightsquigarrow^{\Diamond} 8$
(1) $\mathsf{Trans}(R)$		(9) $b\colon \Diamond_S p$	$6 \rightsquigarrow^{\Diamond} 9$
(2) $R \sqsubseteq S$		(10) $b\colon \Diamond_S c$	$9 \rightsquigarrow^{\Diamond} 10$
(3) $S \sqsubseteq R$		(11) $c\colon p$	$9 \rightsquigarrow^{\Diamond} 11$
(4) $R \sqsubseteq R$	Rel_0	(12) $a\colon \Diamond_R b$	$(8,3) \rightsquigarrow^{\mathsf{Link}} 12$
(5) $S \sqsubseteq S$	Rel_0	(13) $b\colon \Diamond_R c$	$(10,3) \rightsquigarrow^{\mathsf{Link}} 13$
(6) $a\colon \Diamond_S \Diamond_S p$	$0 \rightsquigarrow^{\wedge} 6$	(14) $b\colon \Box_R \neg p$	$(7,12,1,2) \rightsquigarrow^{\mathsf{Trans}} 14$
(7) $a\colon \Box_S \neg p$	$0 \rightsquigarrow^{\wedge} 7$	(15) $c\colon \neg p$	$(14,13) \rightsquigarrow^{\Box} 15$

Next example illustrates the dynamic nature of blockings. Figure 1 represents a complete and open tableau branch $\mathcal{B}$ for the formula $F = (\mathsf{A}{\downarrow}x.\Diamond_{R^-} \Diamond_R \neg x) \wedge \Box_R p$ – which holds in a state w if every state of the interpretation has at least one R-sibling, and p holds in every state R-related to w – where R is a transitive relation. In the representation of the branch given below, $G = \Diamond_{R^-} \Diamond_R \neg x$ and, in the notation $(n, m) \rightsquigarrow^{\mathsf{A}} k$, n is the major premiss of the inference and m the minor one.

The relation $\prec_{\mathcal{B}}$ in this branch can be described as follows, where the notation $n \prec_{\mathcal{B}} \{m_1, \ldots, m_k\}$ abbreviates $n \prec_{\mathcal{B}} m_1$ and $\ldots n \prec_{\mathcal{B}} m_k$. Nodes $0 \ldots 6$ are root nodes, and $6 \prec_{\mathcal{B}} \{7, 8, 9, 12, 16, 18\}$, $8 \prec_{\mathcal{B}} \{10, 11, 13, 14, 15, 17\}$, $17 \prec_{\mathcal{B}} \{19, 20, 23, 33, 35\}$, $18 \prec_{\mathcal{B}} \{21, 22, 26, 31\}$, $20 \prec_{\mathcal{B}} \{24, 25, 29, 30, 32, 34\}$, $22 \prec_{\mathcal{B}} \{27, 28\}$, $35 \prec_{\mathcal{B}} \{36, 37, 38, 41\}$, $37 \prec_{\mathcal{B}} \{39, 40\}$. For instance, node 7 is the minor premiss of the application of the $\Box$ rule producing 12, and 10 is the minor premiss of the application of the Trans rule producing 13, therefore 7 and 12 are siblings and so are 10 and 13. When the A rule is applied to produce node 15 focusing on the nominal a_3, the first non-phantom node where a_3 occurs is 10, so that 10 is the minor premiss of the inference and a sibling of 15.

In order to illustrate blockings, the notation $\mathcal{B}_n$ is used to denote the branch segment up to node n, and $a_i \approx_n a_j$ means that a_i and a_j are compatible in $\mathcal{B}_n$ (note that, in this example, the formulae to be taken into account to check compatibilities are p and $\Box_R p$). Node 17 cannot be blocked by 6, and 20 cannot

0) $a_1 : F$
1) $\mathsf{Trans}(R)$
2) $R \sqsubseteq R$
3) $a_1 : \mathsf{A}{\downarrow}x.G$ $0 \rightsquigarrow^{\wedge} 3$
4) $a_1 : \Box_R p$ $0 \rightsquigarrow^{\wedge} 4$
5) $a_1 : {\downarrow}x.G$ $(3, 0) \rightsquigarrow^{\mathsf{A}} 5$
6) $a_1 : \Diamond_{R^-} \Diamond_R \neg\, a_1$ $5 \rightsquigarrow^{\downarrow} 6$
7) $a_2 : \Diamond_R a_1$ $6 \rightsquigarrow^{\Diamond} 7$
8) $a_2 : \Diamond_R \neg\, a_1$ $6 \rightsquigarrow^{\Diamond} 8$
9) $a_2 : \Box_R p$ $(4, 7, 1, 2) \rightsquigarrow^{\mathsf{Trans}} 9$
10) $a_2 : \Diamond_R a_3$ $8 \rightsquigarrow^{\Diamond} 10$
11) $a_3 : \neg\, a_1$ $8 \rightsquigarrow^{\Diamond} 11$
12) $a_1 : p$ $(9, 7) \rightsquigarrow^{\Box} 12$
13) $a_3 : \Box_R p$ $(9, 10, 1, 2) \rightsquigarrow^{\mathsf{Trans}} 13$
14) $a_3 : p$ $(9, 10) \rightsquigarrow^{\Box} 14$
15) $a_3 : {\downarrow}x.G$ $(3, 10) \rightsquigarrow^{\mathsf{A}} 15$
16) $a_2 : {\downarrow}x.G$ $(3, 7) \rightsquigarrow^{\mathsf{A}} 16$
17) $a_3 : \Diamond_{R^-} \Diamond_R \neg\, a_3$ $15 \rightsquigarrow^{\downarrow} 17$
18) $a_2 : \Diamond_{R^-} \Diamond_R \neg\, a_2$ $16 \rightsquigarrow^{\downarrow} 18$
19) $a_4 : \Diamond_R a_3$ $17 \rightsquigarrow^{\Diamond} 19$
20) $a_4 : \Diamond_R \neg\, a_3$ $17 \rightsquigarrow^{\Diamond} 20$
21) $a_5 : \Diamond_R a_2$ $18 \rightsquigarrow^{\Diamond} 21$
22) $a_5 : \Diamond_R \neg\, a_2$ $18 \rightsquigarrow^{\Diamond} 22$
23) $a_4 : \Box_R p$ $(13, 19, 1, 2) \rightsquigarrow^{\mathsf{Trans}} 23$
24) $a_4 : \Diamond_R a_6$ $20 \rightsquigarrow^{\Diamond} 24$
25) $a_6 : \neg\, a_3$ $20 \rightsquigarrow^{\Diamond} 25$
26) $a_5 : \Box_R p$ $(9, 21, 1, 2) \rightsquigarrow^{\mathsf{Trans}} 26$
27) $a_5 : \Diamond_R a_7$ $22 \rightsquigarrow^{\Diamond} 27$
28) $a_7 : \neg\, a_2$ $22 \rightsquigarrow^{\Diamond} 28$
29) $a_6 : \Box_R p$ $(23, 24, 1, 2) \rightsquigarrow^{\mathsf{Trans}} 29$
30) $a_6 : p$ $(23, 24) \rightsquigarrow^{\Box} 30$
31) $a_2 : p$ $(26, 21) \rightsquigarrow^{\Box} 31$
32) $a_6 : {\downarrow}x.G$ $(3, 24) \rightsquigarrow^{\mathsf{A}} 32$
33) $a_4 : {\downarrow}x.G$ $(3, 19) \rightsquigarrow^{\mathsf{A}} 33$
34) $a_6 : \Diamond_{R^-} \Diamond_R \neg\, a_6$ $32 \rightsquigarrow^{\downarrow} 34$
35) $a_4 : \Diamond_{R^-} \Diamond_R \neg\, a_4$ $33 \rightsquigarrow^{\downarrow} 35$
36) $a_8 : \Diamond_R a_4$ $35 \rightsquigarrow^{\Diamond} 36$
37) $a_8 : \Diamond_R \neg\, a_4$ $35 \rightsquigarrow^{\Diamond} 37$
38) $a_8 : \Box_R p$ $(23, 36, 1, 2) \rightsquigarrow^{\mathsf{Trans}} 38$
39) $a_8 : \Diamond_R a_9$ $37 \rightsquigarrow^{\Diamond} 39$
40) $a_9 : \neg\, a_4$ $37 \rightsquigarrow^{\Diamond} 40$
41) $a_4 : p$ $(38, 36) \rightsquigarrow^{\Box} 41$

Fig. 1. A complete tableau branch for $\{(\mathsf{A}{\downarrow}x.\Diamond_{R^-} \Diamond_R \neg\, x) \wedge \Box_R p, \mathsf{Trans}(R)\}$

be blocked by 8, because a_1 is a top nominal and mappings can only affect non-top ones. In the whole branch $\mathcal{B}$, the nodes 18, 34 and 35 are blocked by 17 (note that 17 is not an ancestor of 18), because $a_3 \approx_{41} a_2 \approx_{41} a_4 \approx_{41} a_6$. Their descendants $(21, 22, 26 - 28, 31, 36 - 41)$ are therefore phantoms in $\mathcal{B}$. However, while 34 is not expanded because it is blocked by 17 in $\mathcal{B}_{34}$ (because $a_3 \approx_{34} a_6$), 18 is not blocked in $\mathcal{B}_i$ for all $i < 31$, i.e. until $a_2 : p$ is added to the branch. Therefore 18 is expanded. Analogously, 35 is not blocked by 17 until $a_4 : p$ is added to the branch (node 41). The branch is complete: every non blocked node has been expanded or used as the minor premiss of a suitable rule. In particular, note that the nominals a_5, a_7, a_8, a_9 occur only in phantom nodes, therefore the A rule cannot focus on them.

3 The Sibyl Prover

The calculus described in Section 2 has been implemented in a prover called Sibyl, that is available at `http://cialdea.dia.uniroma3.it/sibyl/`. It is written in Objective Caml and takes as input a file containing a set of assertions and a set of formulae, checks them for satisfiability and outputs the result. Every input formula in $\mathsf{HL}_m(@, {\downarrow}, \mathsf{E}, \Diamond^-) \setminus \Box{\downarrow}\Box$ is preprocessed and translated into the fragment $\mathsf{HL}_m(@, {\downarrow}, \mathsf{E}, \Diamond^-) \setminus {\downarrow}\Box$, by use of the satisfiability preserving translation defined in [20]. If some formula is not in $\mathsf{HL}_m(@, {\downarrow}, \mathsf{E}, \Diamond^-) \setminus \Box{\downarrow}\Box$, then Sibyl warns the user that termination and correctness of the result are not guaranteed. At

present, backjumping is the only important optimization technique implemented in the prover.

In order to test Sibyl for correctness, it could not be compared to other provers for modal or description logics, since, to the author's knowledge, the hybrid binder and relation hierarchies coexist in none of them. For the same reason it would not make much sense using problems in existing repositories for modal or description logic. Therefore Sibyl has been run on a set of randomly generated tests, and the translations of the same tests into first order logic (using the standard translation of hybrid logic formulae and the straightforward translation of assertions) have then been given in input to the SPASS prover [21]. Each test is based on a file generated by hGen [2], modified so as to obtain formulae in $\mathsf{HL}_m(@, \downarrow, \mathsf{E}, \Diamond^-) \setminus \Box\downarrow\Box$ and with the addition of a random set of transitivity and inclusion assertion. A first group of 1620 tests has been generated with 30% probability for a relation to be transitive and 30% probability for any pair of relations R, S to be related by either $R \sqsubseteq S$ or $R^- \sqsubseteq S$. The tests are grouped according to their modal degree (varying from 2 to 10), each group containing tests with 10 to 50 clauses (hGen generates sets of clauses). In order to evaluate the impact of the presence of assertions on Sibyl's behaviour, other four groups of tests have been obtained from the basic set, reducing the number of assertions in each file, respectively to 75%, 50%, 25% and no assertions at all.

Sibyl and SPASS have been run on these test sets with one minute timeout and they agree on the outcome of all problems where both provers terminate successfully. The test sets, the detailed results of the experiments and diagrams summarizing them can be downloaded from Sibyl web page.

Though the experiments only aimed at testing Sibyl for correctness, they were also an opportunity to give a preliminary evaluations of its performances compared to SPASS (that was run in default mode, since, from some preliminary tests, other flag settings appeared either to degrade its performance or have no significant effect). Quite surprisingly, although SPASS is a mature prover and Sibyl a newborn, the latter turned out to globally outperform the former. SPASS could not solve about 13% of the problems in the allowed one minute time, while Sibyl failed in less than 5.5%. Taking the number of timeouts as a performance measure, the impact of the number of assertions and the modal degree of formulae has been evaluated. In the tests with no assertions SPASS performs better than Sibyl: 2.22% timeouts versus Sibyl's 4.81%. On the other hand, SPASS could not solve 21.98% tests of the base set (with no reduction of the number of assertions), while Sibyl 6.30%. With respect to the effect of the modal degree on the behaviour of the provers, in the base set, for instance, SPASS ran out of time in 2.22% tests of modal degree 2, and it reached 32.22% timeouts in the problems of modal depth 10. In the same set of problems, Sibyl's failures range from 6.67% (modal degree 2) to 9.44% (modal degree 10).

The experimental results show that Sibyl's behaviour only slightly degrades when the number of assertions and the modal degree increase. In comparison, the first order prover appears to be much more sensitive to the number of assertions, especially when the modal degree becomes higher. Presumably, this is not

a credit to Sibyl, but rather an instance of the poor behaviour exhibited by first order theorem provers when fed with non optimized translations of modal formulae. In order to refine such a preliminary analysis, other encoding principles should be used and tested, and the effect of transitivity and inclusion assertions should be analysed separately.

4 Concluding Remarks

This work presents a satisfiability decision procedure for hybrid formulae in $\mathsf{HL}_m(@, \downarrow, \mathsf{E}, \Diamond^-, \mathsf{Trans}, \sqsubseteq) \setminus \Box\downarrow\Box$, and its implementation in the Sibyl prover. Transitivity and relation inclusion assertions are treated by expansion rules which are very close to (though not exactly the same as) the analogous rules presented in [13–16]. The main result of this work is proving that they can be added to a calculus dealing also with restricted occurrences of the binder, maintaining termination, beyond soundness and completeness.

Differently from other terminating tableau calculi for (binder-free) hybrid logic including the global and converse modalities, blocking concerns here nodes (corresponding to formulae) and not nominals (i.e. sets of formulae). In the absence of the binder, compatibility checks, requiring to exit from the "local" view and look for other formulae in the branch, are needed only for the formulae outermost nominals and concern only a subset of the formulae labelled by such nominals. Indirect blocking, in turn, relies on a particular partial order on nodes, arranging them in a family of trees of bounded width and bounded depth. Width boundedness is guaranteed by the fact that universal nodes (which may be expanded a potentially unbounded number of times) do not generate "siblings".

Other works have addressed the issue of representing frame properties and/or relation hierarchies in tableau calculi for binder-free hybrid logic (for instance, [5, 15, 16]). The maybe richer calculus of this kind is [15], that considers graded and global modalities, reflexivity, transitivity and role hierarchies. The converse modalities are however missing, and inverse relations are not allowed.

The possibility of adding graded modalities (i.e. number restrictions of description logics) to the calculus presented in this work is an interesting but hard issue. As a matter of fact, whether restricted occurrences of the binder can coexist with graded modalities in a decidable hybrid logic is an open question.

Acknowledgments. The author's implementation (and debugging) work has built upon the bachelor or master projects of several students. Beyond those who worked on Herod [11], Sibyl's ancestor, the author is especially indebted to Giulia Di Rienzo, who implemented Sibyl's first version.

References

1. Areces, C., Blackburn, P., Marx, M.: A road-map on complexity for hybrid logics. In: Flum, J., Rodríguez-Artalejo, M. (eds.) CSL 1999. LNCS, vol. 1683, pp. 307–321. Springer, Heidelberg (1999)

2. Areces, C., Heguiabehere, J.: hGen: A random CNF formula generator for hybrid languages. In: Methods for Modalities 3 (M4M-3), Nancy, France (2003)
3. Areces, C., ten Cate, B.: Hybrid logics. In: Handbook of Modal Logics, pp. 821–868. Elsevier (2007)
4. Blackburn, P., Seligman, J.: Hybrid languages. Journal of Logic, Language and Information 4, 251–272 (1995)
5. Bolander, T., Blackburn, P.: Terminating tableau calculi for hybrid logics extending K. Electronic Notes in Theoretical Computer Science 231, 21–39 (2009)
6. Cerrito, S., Cialdea Mayer, M.: An efficient approach to nominal equalities in hybrid logic tableaux. Journal of Applied Non-classical Logics 20(1-2), 39–61 (2010)
7. Cerrito, S., Cialdea Mayer, M.: Nominal substitution at work with the global and converse modalities. In: Advances in Modal Logic, vol. 8, pp. 57–74. College Publications (2010)
8. Cerrito, S., Cialdea Mayer, M.: A tableaux based decision procedure for a broad class of hybrid formulae with binders. In: Brünnler, K., Metcalfe, G. (eds.) TABLEAUX 2011. LNCS, vol. 6793, pp. 104–118. Springer, Heidelberg (2011)
9. Cerrito, S., Cialdea Mayer, M.: A tableau based decision procedure for a fragment of hybrid logic with binders. Journal of Automated Reasoning (2012) (published online, to appear on paper)
10. Cialdea Mayer, M.: Tableaux for multi-modal hybrid logic with binders, transitive relations and relation hierarchies. Technical Report RT-DIA-199-2012, Dipartimento di Informatica e Automazione, Università di Roma Tre (2012)
11. Cialdea Mayer, M., Cerrito, S.: Herod and Pilate: two tableau provers for basic hybrid logic. In: Giesl, J., Hähnle, R. (eds.) IJCAR 2010. LNCS, vol. 6173, pp. 255–262. Springer, Heidelberg (2010)
12. Grädel, E.: On the restraining power of guards. Journal of Symbolic Logic 64, 1719–1742 (1998)
13. Horrocks, I., Sattler, U.: A description logic with transitive and inverse roles and role hierarchies. Journal of Logic and Computation 9(3), 385–410 (1999)
14. Horrocks, I., Sattler, U.: A tableau decision procedure for $\mathcal{SHOIQ}$. Journal of Automated Reasoning 39(3), 249–276 (2007)
15. Kaminski, M., Schneider, S., Smolka, G.: Terminating tableaux for graded hybrid logic with global modalities and role hierarchies. Logical Methods in Computer Science 7(1) (2011)
16. Kaminski, M., Smolka, G.: Terminating tableau systems for hybrid logic with difference and converse. Journal of Logic, Language and Information 18(4), 437–464 (2009)
17. Mundhenk, M., Schneider, T.: Undecidability of multi-modal hybrid logics. Electronic Notes in Theoretical Computer Science 174(6), 29–43 (2007)
18. Mundhenk, M., Schneider, T., Schwentick, T., Weber, V.: Complexity of hybrid logics over transitive frames. Journal of Applied Logic 8(4), 422–440 (2010)
19. Szwast, W., Tendera, L.: On the decision problem for the guarded fragment with transitivity. In: Proc. of the 16th Symposium on Logic in Computer Science (LICS), pp. 147–156 (2001)
20. ten Cate, B., Franceschet, M.: On the complexity of hybrid logics with binders. In: Ong, L. (ed.) CSL 2005. LNCS, vol. 3634, pp. 339–354. Springer, Heidelberg (2005)
21. Weidenbach, C., Dimova, D., Fietzke, A., Kumar, R., Suda, M., Wischnewski, P.: SPASS version 3.5. In: Schmidt, R.A. (ed.) CADE 2009. LNCS, vol. 5663, pp. 140–145. Springer, Heidelberg (2009)