A Fixpoint Semantics and an SLD-Resolution Calculus for Modal Logic Programs*

Linh Anh Nguyen

Institute of Informatics
University of Warsaw
ul. Banacha 2, 02-097 Warsaw, Poland
nguyen@mimuw.edu.pl

Abstract

We propose a modal logic programming language called MProlog, which is as expressive as the general modal Horn fragment. We give a fixpoint semantics and an SLD-resolution calculus for MProlog in all of the basic serial modal logics KD, T, KDB, B, $KD4$, $S4$, $KD5$, $KD45$, and $S5$. For an MProlog program P and for L being one of the mentioned logics, we define an operator $T_{L,P}$, which has the least fixpoint $I_{L,P}$. This fixpoint is a set of formulae, which may contain labeled forms of the modal operator $\Diamond$, and is called the least L-model generator of P. The standard model of $I_{L,P}$ is shown to be a least L-model of P. The SLD-resolution calculus for MProlog is designed with a similar style as for classical logic programming. It is sound and complete. We also extend the calculus for MProlog in the almost serial modal logics KB, $K5$, $K45$, and $KB5$.

1 Introduction

Modal and temporal logic programming is a field that extends classical logic programming by adding modal and temporal operators (e.g. $\Box$ and $\Diamond$) to the language in order to reason about belief, knowledge, dynamic changes, etc. A number of systems have been developed for modal and temporal logic programming: Molog [10] is based on modal logics, Pathlog [13] is based on multimodal logics; Templog [1], MetateM [8, 14], Temporal Prolog (Gabbay) [16], and Chronolog [34, 33] are based on temporal logics; Tokio [3], Tempura [27, 18], and Temporal Prolog (Hrycej) [20, 21] are based on interval logics; MTL [9] is based on metric temporal logic.

One of pioneer works on declarative semantics for modal logic programs is the work by Balbiani et al [5]. In that work, the authors gave a declarative semantics and an SLD-resolution calculus for a class of modal logic programs in the logics KD, T, and $S4$. However, there is a restriction on the form of programs (the connective $\Box$ is disallowed in bodies of program clauses and goals), and the SLD-resolution calculus is formulated in a way not close to the style of classical logic programming (as in Lloyd's book [24]). Akama [2] and Debart et al [13] applied a technique of translation to classical logic for (multi)modal logic programming. The technique is similar to the one used in Ohlbach's resolution calculus for modal logics [31]. Extra parameters are added to function symbols and predicate symbols to represent paths in the Kripke model, and special unification algorithms are developed to deal with them. The SLD E-resolution given in [13] is quite efficient, while the minimal model semantics is less declarative. The minimal Herbrand model defined in [13] for a program is not a Kripke model, and it is not shown how a minimal Kripke model can be constructed from the minimal Herbrand model.

The goal of this work is to develop the least model semantics, a fixpoint semantics, and an SLD-resolution calculus, in a "direct" way and closely to the style of classical logic programming,

*This is a revised version of "L.A. Nguyen. *A Fixpoint Semantics and an SLD-Resolution Calculus for Modal Logic Programs*. Fundamenta Informaticae, 55(1), 2003, 63-100."

for modal logic programs in the basic serial modal logics KD, T, KDB, B, $KD4$, $S4$, $KD5$, $KD45$, and $S5$. We will also extend the SLD-resolution calculus for modal logic programs in the almost serial modal logics KB, $K5$, $K45$, and $KB5$. We do not require any restriction on occurrences of $\Box$ and $\Diamond$ in program clauses.

To illustrate the idea of this work, let us consider an example. Let the base logic be KD, $G = \leftarrow \Box p_4(x)$ be a goal and P a program that consists of the following formulae:

$$\begin{array}{l}\phi_1 = \Diamond p_1(a) \leftarrow \\ \phi_2 = \Box(\, \Box p_2(x) \leftarrow p_1(x)\,) \\ \phi_3 = \Box(\, p_3(x) \leftarrow p_1(x), \Box p_2(x)\,) \\ \phi_4 = \Box p_4(x) \leftarrow \Diamond p_3(x)\end{array}$$

When building a KD-model graph M for P, to realize ϕ_1 at the actual world τ we connect τ to a world w and add $p_1(a)$ to w. Thus w can be identified by τ and the atom $p_1(a)$. At the same time we want to represent the model corresponding to M by a set I of *atoms*. To keep the information that $p_1(a)$ is true at w, we add the atom $\langle p_1(a)\rangle\, p_1(a)$ to I, where $\langle p_1(a)\rangle$ is $\Diamond$ labeled by $p_1(a)$. To realize ϕ_2 at τ, $\Box p_2(x) \leftarrow p_1(x)$ is added to w, and then $\Box p_2(a)$ is also added to w. To keep the fact that $\Box p_2(a)$ belongs to w, we add $\langle p_1(a)\rangle \Box p_2(a)$ to I. Now take another view, just leave M and concentrate only on I. Note that I contains $\langle p_1(a)\rangle\, p_1(a)$ and $\langle p_1(a)\rangle \Box p_2(a)$. Apply the rule ϕ_3 to I, then I should contain also $\langle p_1(a)\rangle\, p_3(a)$. Apply the rule ϕ_4 to I, then I should contain also $\Box p_4(a)$. In general, instead of building a model graph for P we can build such a set I of atoms, which is called a *model generator*. The set $I_{KD,P} = \{\langle p_1(a)\rangle\, p_1(a), \langle p_1(a)\rangle \Box p_2(a), \langle p_1(a)\rangle\, p_3(a), \Box p_4(a)\}$ is the least set of ground atoms which can be derived from P in KD. This set is obtained as the least fixpoint of a certain operator $T_{KD,P}$ and is called the *least KD-model generator* of P.

Given a model generator I, we can construct the *standard KD-model* for it by building a model graph. During the construction, to realize a formula $\langle E\rangle\phi$ at a world w, where E is a ground classical atom, we connect w to the world identified by w and E (in the form $w\langle E\rangle$) and add ϕ to that world. We realize a formula $\Box\phi$ at a world w by adding ϕ to every world reachable from w. To guarantee the constructed model graph to be the smallest, each new world is connected to an empty world at the time of its creation. It can be shown that the standard model of $I_{KD,P}$ is a least KD-model of P.

Now let us give an SLD-refutation of $P \cup \{G\}$ in KD. By the content of $I_{KD,P}$, the computed answer should be $\{x/a\}$. The SLD-refutation should trace back the process of deriving the atom $\Box p_4(a)$ of $I_{KD,P}$ from P. As a KD-resolvent of G and ϕ_4, we derive a new goal $G_1 = \leftarrow \Diamond p_3(x)$. As a KD-resolvent of G_1 and ϕ_3, we derive the goal $G_2 = \leftarrow \Diamond(p_1(x) \wedge \Box p_2(x))$. This goal is not desired, as it contains a formula but not atoms in its body. To overcome this problem, the goal atom $\Diamond p_3(x)$ in G_1 is first labeled as $\langle X\rangle\, p_3(x)$, where X is a fresh variable for classical atoms. Suppose G_1 is transformed to $G_1' = \leftarrow \langle X\rangle\, p_3(x)$ and G_2 is now a KD-resolvent of G_1' and ϕ_3. Then $G_2 = \leftarrow \langle X\rangle\, p_1(x), \langle X\rangle \Box p_2(x)$. Now resolve G_2 with ϕ_1. As explained in the construction of $I_{KD,P}$, the atom $\Diamond p_1(a)$ in the head of ϕ_1 can be treated as $\langle p_1(a)\rangle\, p_1(a)$. Thus, resolving G_2 with ϕ_1 results in $G_3 = \leftarrow \langle p_1(a)\rangle \Box p_2(a)$ and an mgu $\{x/a, X/p_1(a)\}$. Resolving G_3 with ϕ_2, we obtain $G_4 = \leftarrow \langle p_1(a)\rangle\, p_1(a)$. Finally, resolving G_4 with ϕ_1, we obtain the empty clause (denoted by $\diamond$). Note that for simplicity we have ignored renaming variables. The refutation is summarized in the following table.

Goals	Input Clauses	Mgu's
$G = \leftarrow \Box p_4(x)$		
$G_1 = \leftarrow \Diamond p_3(x)$	ϕ_4	ε
$G_1' = \leftarrow \langle X\rangle\, p_3(x)$		
$G_2 = \leftarrow \langle X\rangle\, p_1(x), \langle X\rangle \Box p_2(x)$	ϕ_3	ε
$G_3 = \leftarrow \langle p_1(a)\rangle \Box p_2(a)$	ϕ_1	$\{x/a, X/p_1(a)\}$
$G_4 = \leftarrow \langle p_1(a)\rangle\, p_1(a)$	ϕ_2	ε
$G_5 = \diamond$	ϕ_1	ε

In this work, we propose a modal logic programming language called MProlog, which is as expressive as the general modal Horn fragment. We give a fixpoint semantics and an SLD-resolution

calculus for MProlog in all of the basic serial modal logics KD, T, KDB, B, $KD4$, $S4$, $KD5$, $KD45$, and $S5$. For an MProlog program P and for L being one of the mentioned logics, we define an operator $T_{L,P}$, which has the least fixpoint $I_{L,P}$. This fixpoint is a set of formulae, which may contain labeled forms of the modal operator $\Diamond$, and is called the *least L-model generator* of P. The *standard model* of $I_{L,P}$ is shown to be a least L-model of P. The SLD-resolution calculus for MProlog is designed with a similar style as for classical logic programming. It is sound and complete. We also extend the calculus for MProlog in the almost serial modal logics KB, $K5$, $K45$, and $KB5$.

This work is organized as follows: In Section 2 we give basic definitions for fixed-domain first-order modal logics with rigid terms. MProlog programs are interpreted in such logics. In that section we also introduce labeled modal operators and their semantics. In Section 3 we define the MProlog language and show that it is as expressive as the general modal Horn fragment. In Section 4 we study model generators and their standard models. The fixpoint semantics and the least model semantics for MProlog programs are given in Section 5, while the SLD-resolution calculus for MProlog is given in Section 6. Soundness and completeness of the calculus is proved in Section 7. In Section 8 we extend the SLD-resolution calculus for MProlog programs in the almost serial modal logics KB, $K5$, $K45$, and $KB5$. Some related works are discussed in Section 9. Section 10 concludes this work.

2 Preliminaries

In this section, we first give definitions for fixed-domain first-order modal logics with rigid terms. We then define an order between Kripke models. In the last subsection, we give a list of notations that will be used in this work and introduce labeled modal operators.

2.1 First Order Modal Logics

An *alphabet* for modal logics consists of variables, constant symbols, function symbols, predicate symbols, the classical connectives $\wedge$, $\vee$, $\neg$, $\rightarrow$, the modal operators $\Box$, $\Diamond$, the quantifiers $\forall$, $\exists$, and the punctuation symbols "(", ")", ",".

The symbols $\neg$, $\wedge$, $\vee$ and $\rightarrow$, respectively, stand for logical negation, logical conjunction, logical disjunction and logical implication. The symbols $\Box$ and $\Diamond$ can take various meanings but traditionally stand for "necessity" and "possibility". The symbol $\forall$ is called the universal quantifier and $\exists$ the existential quantifier.

To enable us to omit parentheses, we adopt the convention that the connectives $\neg$, $\Box$, $\Diamond$ and the quantifiers $\forall$, $\exists$ are of equal binding strength but bind stronger than $\wedge$, which binds stronger than $\vee$, which binds stronger than $\rightarrow$.

Definition 2.1 A *term* is defined inductively as follows: a variable is a term; a constant symbol is a term; if f is an n-ary function symbol and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term.

Definition 2.2 A *(well-formed modal) formula* is defined inductively as follows:

- If p is an n-ary predicate symbol and $t_1, \ldots, t_n$ are terms, then $p(t_1, \ldots, t_n)$ is a formula, called a *classical atom*.
- If ϕ and ψ are formulae, then so are $(\neg\phi)$, $(\phi \wedge \psi)$, $(\phi \vee \psi)$, $(\phi \rightarrow \psi)$, $(\Box\phi)$, and $(\Diamond\psi)$.
- If ϕ is a formula and x is a variable, then $(\forall x\ \phi)$ and $(\exists x\ \phi)$ are formulae.

We also write $\phi \equiv \psi$ for $(\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi)$.

The *modal depth* of a formula ϕ is the maximal nesting depth of modalities occurring in ϕ. For example, the modal depth of $\Box(\Diamond p(x) \vee \Box q(y))$ is 2.

The *scope* of $\forall x$ (resp. $\exists x$) in $\forall x\ \phi$ (resp. $\exists x\ \phi$) is ϕ. A *bound occurrence* of a variable in a formula is an occurrence immediately following a quantifier or an occurrence within the scope of a quantifier, which has the same variable immediately after the quantifier. Any other occurrence of a variable is *free*.

A *closed formula* is a formula without free occurrences of any variable. If ϕ is a formula, then $\forall(\phi)$ denotes the *universal closure* of ϕ, which is the closed formula obtained by adding a universal quantifier for every variable having a free occurrence in ϕ. Similarly, $\exists(\phi)$ denotes the *existential closure* of ϕ, which is obtained by adding an existential quantifier for every variable having a free occurrence in ϕ.

A *ground term* is a term without variables. A *ground formula* is a formula without quantifiers and variables. The *Herbrand universe* $\mathcal{U}$ is the set of all ground terms. The *Herbrand base* $\mathcal{B}$ is the set of all ground classical atoms.

We now define Kripke models, model graphs, and the satisfaction relation.

Definition 2.3 A *Kripke frame* is a triple $\langle W, \tau, R\rangle$, where W is a nonempty set of possible worlds, $\tau \in W$ is the *actual world*, and R is a binary relation on W, called the *accessibility relation*. If $R(w, u)$ holds then we say that the world u *is accessible from the world* w, or that u *is reachable from* w.

Definition 2.4 A *fixed-domain Kripke model with rigid terms*, hereafter simply called a Kripke model or just a model, is a tuple $M = \langle D, W, \tau, R, m\rangle$, where D is a set called *domain*, $\langle W, \tau, R\rangle$ is a Kripke frame, and m is an interpretation of constant symbols, function symbols and predicate symbols. For a constant symbol a, $m(a)$ is an element of D, denoted also by a^M. For an n-ary function symbol f, $m(f)$ is a function from D^n to D, denoted also by f^M. For an n-ary predicate symbol p and a world $w \in W$, $m(w)(p)$ is a n-ary relation on D, denoted also by $p^{M,w}$.

Definition 2.5 A *model graph* is a tuple $\langle W, \tau, R, H\rangle$, where $\langle W, \tau, R\rangle$ is a Kripke frame and H is a function that maps each world of W to a set of formulae.

Every model graph $\langle W, \tau, R, H\rangle$ corresponds to a Herbrand model $M = \langle \mathcal{U}, W, \tau, R, m\rangle$ specified by: $c^M = c$, $f^M(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$, and $p^{M,w}(t_1, \ldots, t_n) \equiv (p(t_1, \ldots, t_n) \in H(w))$, where $t_1, \ldots, t_n$ are ground terms. We will sometimes treat a model graph as its corresponding model.

Definition 2.6 Let M be a Kripke model. A *variable assignment* (w.r.t. M) is a function that maps each variable to an element of the domain of M. The value of a term t w.r.t. a variable assignment V is denoted by $V(t)$ and defined as follows: If t is a constant symbol a then $V(t) = a^M$; if t is a variable x then $V(t) = V(x)$; if t is $f(t_1, \ldots, t_n)$ then $V(t) = f^M(V(t_1), \ldots, V(t_n))$.

Definition 2.7 Given some Kripke model $M = \langle D, W, \tau, R, m\rangle$, some variable assignment V, and some world $w \in W$, the *satisfaction relation* $M, V, w \vDash \zeta$ for a formula ζ is defined as follows:

$M, V, w \vDash p(t_1, \ldots, t_n)$ iff $p^{M,w}(V(t_1), \ldots, V(t_n))$;
$M, V, w \vDash \neg\phi$ iff $M, V, w \nvDash \phi$;
$M, V, w \vDash \phi \wedge \psi$ iff $M, V, w \vDash \phi$ and $M, V, w \vDash \psi$;
$M, V, w \vDash \phi \vee \psi$ iff $M, V, w \vDash \phi$ or $M, V, w \vDash \psi$;
$M, V, w \vDash \phi \rightarrow \psi$ iff $M, V, w \nvDash \phi$ or $M, V, w \vDash \psi$;
$M, V, w \vDash \Box\phi$ iff for all $v \in W$ such that $R(w, v)$, $M, V, v \vDash \phi$;
$M, V, w \vDash \Diamond\phi$ iff there exists some $v \in W$ such that $R(w, v)$ and $M, V, v \vDash \phi$;
$M, V, w \vDash \forall x\ \phi$ iff for all $a \in D$, $(M, V', w \vDash \phi)$,
where $V'(x) = a$ and $V'(y) = V(y)$ for $y \neq x$;
$M, V, w \vDash \exists x\ \phi$ iff there exists $a \in D$ such that $M, V', w \vDash \phi$,
where $V'(x) = a$ and $V'(y) = V(y)$ for $y \neq x$.

If $M, V, w \vDash \phi$ then we say that ϕ is true at w in M w.r.t. V. We write $M, w \vDash \phi$ to denote that $M, V, w \vDash \phi$ for every V. We say that M satisfies ϕ, or ϕ is true in M, and write $M \vDash \phi$, if $M, \tau \vDash \phi$. For a set Γ of formulae, we call M a model of Γ and write $M \vDash \Gamma$ if $M \vDash \alpha$ for every $\alpha \in \Gamma$.

A *logic* can be defined by a set of well-formed formulae, a class of admissible interpretations, and a satisfaction relation. The class of admissible interpretations for a modal logic L is often specified by restrictions on Kripke frames admissible for L. We refer to such restrictions by *L-frame restrictions* and call frames with such properties *L-frames*.

Definition 2.8 We call a model M with L-frame an *L-model*. We say that ϕ is *L-satisfiable* if there exists an L-model of ϕ, i.e. an L-model satisfying ϕ. A formula ϕ is said to be *L-valid* and called an *L-tautology* if ϕ is true in every L-model. For a set Γ of formulae, we write $\Gamma \models_L \phi$ and call ϕ a *logical consequence* of Γ in L if ϕ is true in every L-model of Γ.

If we define the class of admissible interpretations to be the class of all Kripke models (without restrictions on the accessibility relations) then we obtain the fixed-domain first-order modal logic with rigid terms K. This logic is axiomatized by the following system:

- axioms for the classical predicate logic (without identity)
- the K-axiom: $\Box(\phi \to \psi) \to (\Box\phi \to \Box\psi)$
- the Barcan formula axiom: $\forall x\, \Box\phi \to \Box\, \forall x\, \phi$
- the axiom defining $\Diamond$: $\Diamond\phi \equiv \neg\Box\neg\phi$
- the modus ponens rule: $\dfrac{\phi \quad \phi \to \psi}{\psi}$
- the generalization rule: $\dfrac{\phi}{\forall x\, \phi}$
- and the modal generalization rule: $\dfrac{\phi}{\Box\phi}$

Note that the converse Barcan formula $\Box\forall x\, \phi \to \forall x\, \Box\phi$ is a consequence of this axiomatization system. It is known that this system is sound and complete.

Every logic whose axiomatization is an extension of the K system is called a *normal modal logic*. Some additional axioms like D, T, B, 4, 5 given in Table 2.1 correspond to certain conditions on the accessibility relation in the sense that by adding some of them to the system K we obtain an axiomatization system which is sound and complete with respect to the class of admissible interpretations that satisfy the corresponding restrictions on the accessibility relation. Some of the most popular normal modal logics together with their additional axioms and frame restrictions are listed in Table 2.1. For a survey on axiomatization systems for first-order modal logics and methods for proving their soundness and completeness, we refer the reader to Garson's work [17].

2.2 Ordering Kripke Models

A formula is in the *negative normal form* if it does not contain the connective $\to$, and each of its negations occurs immediately before a classical atom. Every formula can be transformed to the equivalent negative normal form in the usual way.

Definition 2.9 A formula is called *positive* if its negative normal form does not contain negation. A formula is called *negative* if its negation is a positive formula.

Definition 2.10 A model M is said to be *less than* or *equal to* N, write $M \leq N$, if for any positive ground formula ϕ, if M satisfies ϕ then N also satisfies ϕ.

The relation $\leq$ in the above definition is a pre-order[1].

Definition 2.11 Let $M = \langle D, W, \tau, R, m\rangle$ and $N = \langle D', W', \tau', R', m'\rangle$ be Kripke models. We say that M is *less than or equal to N w.r.t. a binary relation* $r \subseteq W \times W'$, and write $M \leq_r N$, if the following conditions hold:

1. $r(\tau, \tau')$
2. $\forall x, x', y \;\; R(x, y) \wedge r(x, x') \to \exists y' \; R'(x', y') \wedge r(y, y')$
3. $\forall x, x', y' \;\; R'(x', y') \wedge r(x, x') \to \exists y \; R(x, y) \wedge r(y, y')$.

[1] i.e. a reflexive and transitive binary relation

Axiom	Schema	Corresponding Condition on R
D	$\Box\phi \rightarrow \Diamond\phi$	$\forall w\ \exists u\ R(w,u)$
T	$\Box\phi \rightarrow \phi$	$\forall w\ R(w,w)$
B	$\phi \rightarrow \Box\Diamond\phi$	$\forall w,u\ R(w,u) \rightarrow R(u,w)$
4	$\Box\phi \rightarrow \Box\Box\phi$	$\forall w,u,v\ R(w,u) \wedge R(u,v) \rightarrow R(w,v)$
5	$\Diamond\phi \rightarrow \Box\Diamond\phi$	$\forall w,u,v\ R(w,u) \wedge R(w,v) \rightarrow R(u,v)$

Logic	Axioms	Frame Restriction
K		no restriction
KD	D	serial
T	T	reflexive
KB	B	symmetric
KDB	DB	serial and symmetric
B	TB	reflexive and symmetric
$K4$	4	transitive
$KD4$	D4	serial and transitive
$S4$	T4	reflexive and transitive
$K5$	5	euclidean
$KD5$	D5	serial and euclidean
$K45$	45	transitive and euclidean
$KD45$	D45	serial, transitive and euclidean
$KB5$	B5	symmetric and euclidean
$S5$	T5	reflexive and euclidean

Table 1: Modal logics and frame restrictions

4. For any $x \in W$, $x' \in W'$ such that $r(x,x')$, for any ground classical atom E, if $M,x \vDash E$ then $N,x' \vDash E$.

In the above definition, the first three conditions state that r is a bisimulation of the frames of M and N. Intuitively, $r(x,x')$ states that the world x is less than or equal to x'.

Lemma 2.1 *If $M \leq_r N$ then $M \leq N$.*

The proof of this lemma is straightforward; see [29] for the propositional case.

2.3 Notations, Labeled Modal Operators, and Unification

Throughout this work, we will use the following notations:

- $\top$ – the *truth* symbol, with the usual[2] semantics;
- D, E, F – classical atoms or $\top$;
- X, Y – variables for classical atoms or $\top$, called *atom variables*;
- $\langle E\rangle$ – $\Diamond$ labeled by E;
- $\langle X\rangle$ – $\Diamond$ labeled by X;
- ∇ – $\Box$, $\Diamond$, $\langle E\rangle$, or $\langle X\rangle$, called a modal operator;
- $\triangle$ – a sequence of modal operators, which can be empty and is called a *modality*;
- A, B – formulae of the form E or ∇E, called *simple atoms*;
- α, β – formulae of the form $\triangle E$, called *atoms*;

[2] i.e. it is always true that $M,V,w \vDash \top$

- ϕ, ψ – *labeled formulae* (i.e. formulae that may contain $\langle E\rangle$ and $\langle X\rangle$), which will be simply called formulae.

A *ground formula* is redefined to be a formula without variables and atom variables. A *closed formula* is redefined to be a formula without atom variables and free occurrences of variables. A *labeled modal operator* is a modal operator of the form $\langle E\rangle$ or $\langle X\rangle$. A modal operator is said to be ground if it is $\Box$, $\Diamond$, or $\langle E\rangle$ with E being $\top$ or a ground classical atom. A *ground modality* is a modality that contains only ground modal operators.

Denote $EdgeLabels = \{\langle E\rangle \mid E \in \mathcal{B} \cup \{\top\}\}$. The semantics of $\langle E\rangle$ for E being $\top$ or a ground classical atom is specified by the following definition.

Definition 2.12 Let $M = \langle D, W, \tau, R, m\rangle$ be a Kripke model. A *$\Diamond$-realization function on M* is a partial function $\sigma : W \times EdgeLabels \to W$ such that if $\sigma(w, \langle E\rangle) = u$, then $R(w, u)$ holds and $M, u \vDash E$. Given a $\Diamond$-realization function σ, a world $w \in W$, and a ground (labeled) formula ϕ, the satisfiability relation $M, \sigma, w \vDash \phi$ is defined in the usual way, except that $M, \sigma, w \vDash \langle E\rangle\psi$ iff $\sigma(w, \langle E\rangle)$ is defined and $M, \sigma, \sigma(w, \langle E\rangle) \vDash \psi$. We write $M, \sigma \vDash \phi$ to denote that $M, \sigma, \tau \vDash \phi$. For a set of ground atoms I, we write $M, \sigma \vDash I$ to denote that $M, \sigma \vDash \alpha$ for all $\alpha \in I$. We write $M \vDash I$, and call M a model of I, if $M, \sigma \vDash I$ for *some* σ.

We now give definitions for substitution and unification.

Definition 2.13 A *substitution* θ is a (finite or infinite) set of the form $\{x_1/t_1, x_2/t_2, \ldots, X_1/E_1, X_2/E_2, \ldots, Y_1/Z_1, Y_2/Z_2, \ldots\}$, where $x_1, x_2, \ldots$ are distinct variables, $t_1, t_2, \ldots$ are terms, $X_1, X_2, \ldots, Y_1, Y_2, \ldots$ are distinct atom variables, and for any element v/s of the set, s is distinct from v. The set $\{x_1, x_2, \ldots, X_1, X_2, \ldots, Y_1, Y_2, \ldots\}$ is called the *domain* of θ and denoted by $Dom(\theta)$. θ is called a *ground substitution* if the set $\{Y_1, Y_2, \ldots\}$ is empty, $t_1, t_2, \ldots$ are ground terms, and $E_1, E_2, \ldots$ are ground classical atoms. The substitution given by the empty set is called the *identity substitution* and denoted by ε.

An *expression* is either a term or a formula. Let $\theta = \{v_1/s_1, v_2/s_2, \ldots\}$ be a substitution (where v_i/s_i is of the form x/t, X/E, or Y/Z) and Φ be an expression. Then $\Phi\theta$, the *instance* of Φ by θ, is the expression obtained from Φ by simultaneously replacing each occurrence of v_i in Φ by s_i. For example, if $\Phi = p(x, y, a)$ and $\theta = \{x/y,\ y/f(x, b)\}$, then $\Phi\theta = p(y, f(x, b), a)$. If S is a set of expressions, then by $S\theta$ we denote the set $\{\Phi\theta \mid \Phi \in S\}$.

Let θ and γ be substitutions. Let $\delta = \{v/(s\gamma) \mid v/s \in \theta\} \cup \{v/s \mid v/s \in \gamma \text{ and } v \notin dom(\theta)\}$. Then the *composition* $\theta\gamma$ is the substitution obtained from δ by deleting every element v/s with $s = v$. Here are some properties of substitutions: if θ, γ, δ are substitutions and Φ is an expression, then $\theta\varepsilon = \varepsilon\theta = \theta$, $(\Phi\theta)\gamma = \Phi(\theta\gamma)$, and $(\theta\gamma)\delta = \theta(\gamma\delta)$.

Definition 2.14 Let Φ and Ψ be expressions. We say that Φ is a *variant* of Ψ if there exist substitutions θ and γ such that $\Phi = \Psi\theta$ and $\Psi = \Phi\gamma$.

Definition 2.15 Let S be a finite set of expressions. A substitution θ is called a *unifier* for S if $S\theta$ is singleton. A unifier θ for S is called a *most general unifier* (mgu) for S if, for each unifier γ of S, there exists a substitution δ such that $\gamma = \theta\delta$. If there exists a unifier for S then S is said to be *unifiable*.

The well-known unification theorem says that if S is a finite unifiable set of expressions, then there exists an mgu for S, which can be effectively computed (see e.g. [24]).

Atom variables in modal operators of the form $\langle X\rangle$ are mainly interpreted by substitutions. When a formula ϕ is taken to be semantically considered, all modal operators $\langle X\rangle$ in ϕ are treated as[3] $\langle\top\rangle$, which is formalized by the following definition.

[3]Atom variables appear only in goal bodies. In the negation of a goal (i.e. a query) they are existentially quantified. Hence it is sufficient to choose some concrete values for them. Furthermore, as we will see, the modal operator $\langle\top\rangle$ plays the role of $\Box$; and if X remains at the end as an unsubstituted atom variable then $\langle X\rangle$ intuitively also plays the role of $\Box$.

Definition 2.16 Given a Kripke model M, a $\Diamond$-realization function σ, and a labeled formula ϕ without quantifiers, we write $M, \sigma \vDash \forall_c(\phi)$ to denote that for any substitution θ which substitutes every variable by a ground term and does not substitute atom variables, $M, \sigma \vDash \phi\,\theta\,\delta_\top$, where $\delta_\top = \{X/\top \mid X \text{ is an atom variable}\}$. By $M \vDash \forall_c(\phi)$ we denote $M, \sigma \vDash \forall_c(\phi)$ for *some* σ. For a formula set Γ and a formula ϕ (maybe in the form $\forall_c(\psi)$), we write $\Gamma \vDash_L \phi$ to denote that for any L-model M, if $M \vDash \Gamma$ then $M \vDash \phi$.

The quantifier $\forall_c$ is introduced because $\Diamond$-realization functions are defined using Herbrand base and we do not want to restrict only to Herbrand models. Let L be one of the logics given in Table 2.1. Suppose that there are enough constant symbols not occurring in Γ, for example, infinitely many. Then, because L has a complete axiomatization, for Γ being a formula set and ϕ a formula - both without labeled modal operators, $\Gamma \vDash_L \forall(\phi)$ iff $\Gamma \vDash_L \forall_c(\phi)$.

3 The MProlog Language

In this section we define a class of modal logic programs and goals, which specify a language called *MProlog*. Despite its simple form, the language is as expressive as its extension called *eMProlog*. This section is free from labeled modal operators, hereby we assume that formulae in this section do not contain labeled modal operators.

In logic programming, it is convenient to adopt a special clausal notation. Throughout, by

$$\Box^s(\phi_1, \ldots, \phi_m \leftarrow \psi_1, \ldots, \psi_n)$$

we denote the formula

$$\forall x_1 \ldots \forall x_k\ \Box^s(\phi_1 \vee \ldots \vee \phi_m \vee \neg\psi_1 \vee \ldots \vee \neg\psi_n)$$

where $s \geq 0$ and $\Box^s$ is a sequence with length s of $\Box$, $\phi_1, \ldots, \phi_m, \psi_1, \ldots, \psi_n$ are positive formulae not containing quantifiers, and $x_1, \ldots, x_k$ are all variables occurring in the formula. Thus, in the clausal notation, all variables are assumed to be universally quantified, the commas in the antecedent $\psi_1, \ldots, \psi_n$ denote conjunction and the commas in the consequent $\phi_1, \ldots, \phi_m$ denote disjunction.

Definition 3.1 A *program clause* is a formula of the form

$$\Box^s(A \leftarrow B_1, \ldots, B_n)$$

where $s \geq 0$, $n \geq 0$, and A, B_1, $\ldots$, B_n are formulae of the form E, $\Box E$, or $\Diamond E$ with E being a classical atom. $\Box^s$ is called the *modal context*, A is called the *head*, and $B_1, \ldots, B_n$ is called the *body* of the program clause.

Definition 3.2 A *unit clause* is a clause of the form $\Box^s(A \leftarrow)$, i.e. a program clause with an empty body. The clause with empty head and empty body is called the *empty clause* and denoted by $\diamond$.

Definition 3.3 An *MProlog program* is a finite set of program clauses[4].

Definition 3.4 An *MProlog goal atom* is a formula of the form $\Box^s E$ or $\Box^s\Diamond E$, where $s \geq 0$. An *MProlog query* is a formula of the form $\exists(\alpha_1 \wedge \ldots \wedge \alpha_k)$, where α_1, $\ldots$, α_k are MProlog goal atoms. An *MProlog goal* is the negation of an MProlog query.

In the clausal notation, the goal $\neg\exists(\alpha_1 \wedge \ldots \wedge \alpha_k)$ is presented as $\leftarrow \alpha_1, \ldots, \alpha_k$. If P is an MProlog program, $Q = \exists(\alpha_1 \wedge \ldots \wedge \alpha_k)$ is an MProlog query and $G = \leftarrow \alpha_1, \ldots, \alpha_k$ is the corresponding goal, then $P \vDash_L Q$ iff $P \cup \{G\}$ is L-unsatisfiable. For the proof of this statement, just note that $G = \forall(\neg(\alpha_1 \wedge \ldots \wedge \alpha_k))$.

In the logic $KD5$, we have a tautology $\triangle\nabla_1\nabla_2\phi \equiv \nabla_1\nabla_2\phi$. This equivalence was given in [12]. In $KD5$, we also have $\Diamond\nabla\phi \equiv \Box\nabla\phi$. Furthermore, in the logics $KD45$ and $S5$, we have $\nabla_1\nabla_2\phi \equiv \nabla_2\phi$. So, for these logics we refine the definitions of MProlog programs and goals as follows.

[4]For the logics $KD5$, $KD45$, and $S5$, we will adopt some restrictions on the form of program clauses.

Definition 3.5 Let L be one of the logics $KD5$, $KD45$, $S5$. An *MProlog program in* L is a finite set of program clauses of the form

$$\Box^s(A \leftarrow B_1, \ldots, B_n)$$

where $s \leq 2$ for $L = KD5$, and $s \leq 1$ for $L \in \{KD45, S5\}$. An *MProlog goal atom in* L is a formula of the form $\Box^k E$ or $\Box^h \Diamond E$, where $0 \leq k \leq 2$ and $0 \leq h \leq 1$ for $L = KD5$, and $0 \leq k \leq 1$ and $h = 0$ for $L \in \{KD45, S5\}$. MProlog queries and MProlog goals in L are defined as usual.

When the base logic is one of $KD5$, $KD45$, and $S5$, we implicitly adopt the above-mentioned conditions for MProlog programs and goals.

We now give an extension of MProlog, which is called eMProlog and stands for the general modal Horn fragment.

Definition 3.6 A formula ϕ without quantifiers is called a *non-negative modal Horn formula (without quantifiers)* if one of the following conditions holds:

- ϕ is a classical atom;
- $\phi = \psi \leftarrow \zeta$, where ψ is a non-negative modal Horn formula and ζ is a positive formula (in the negative normal form);
- $\phi = \Box\psi$, or $\phi = \Diamond\psi$, or $\phi = \psi \wedge \zeta$, where ψ and ζ are non-negative modal Horn formulae.

Definition 3.7 An *eMProlog program* is a finite set of formulae of the form $\forall(\phi)$, where ϕ is a non-negative modal Horn formula without quantifiers. An *eMProlog query* is a formula of the form $\exists(\phi)$, where ϕ is a positive formula without quantifiers. An *eMProlog goal* is the negation of an eMProlog query.

To show that MProlog is as expressive as eMProlog, we first define correct answers.

Definition 3.8 Let P be an MProlog (resp. eMProlog) program and G an MProlog (resp. eMProlog) goal. An *answer* for $P \cup \{G\}$ is a substitution for variables of G (i.e. a substitution θ such that $Dom(\theta)$ is a set of variables occurring in G).

Definition 3.9 Let L be a modal logic, P an MProlog (resp. eMProlog) program, $Q = \exists(\phi)$ an MProlog (resp. eMProlog) query and G the corresponding goal (i.e. $G = \neg Q$). Let θ be an answer for $P \cup \{G\}$. We say that θ is a *correct answer* in L for $P \cup \{G\}$ if $P \models_L \forall(\phi\,\theta)$.

The following proposition states that MProlog has the same expressiveness as eMProlog.

Proposition 3.1 *For any eMProlog program P and any eMProlog goal G, there exist an MProlog program P' and an MProlog goal G' such that every correct answer in L for $P \cup \{G\}$ is a correct answer in L for $P' \cup \{G'\}$ and vice versa.*

Proof. For any formula ϕ with n variables and no quantifiers, by p_ϕ we denote a fresh n-ary predicate symbol, and by E_ϕ we denote the classical atom $p_\phi(x_1, \ldots, x_n)$, where $x_1, \ldots, x_n$ are the variables occurring in ϕ. In this proof we will use Γ and Δ to denote sequences of formulae.

Let P be an eMProlog program and $G = \leftarrow \xi$ an eMProlog goal. Let $P'' = P \cup \{E_\xi \leftarrow \xi\}$ and $G' = \leftarrow E_\xi$. We apply Mints translation[5] to P'' in order to obtain an MProlog program. The translation rules are given in the below list, in which each row contains a formula to be replaced in the left, and the replacing ones in the right. The formula ϕ in the rules (1), (3) and (4), and ψ in the rules (7) and (8) are required not to be a classical atom. The rules (5) and (9) are designed

[5]This translation technique for modal logics was probably used the first time in the Mints' work [26].

to shorten the result. We apply the translation rules to P'' until no further changes can be made. Let P' be the resulting set.

$$\forall(\Box^k\Diamond\phi) \qquad \forall(\Box^k\Diamond E_\phi),\ \forall(\Box^{k+1}(\phi \leftarrow E_\phi)) \tag{1}$$
$$\forall(\Box^k(\phi \wedge \psi)) \qquad \forall(\Box^k\phi),\ \forall(\Box^k\psi) \tag{2}$$
$$\forall(\Box^k(\Box\phi \leftarrow \Gamma)) \qquad \forall(\Box^k(\Box E_\phi \leftarrow \Gamma)),\ \forall(\Box^{k+1}(\phi \leftarrow E_\phi)) \tag{3}$$
$$\forall(\Box^k(\Diamond\phi \leftarrow \Gamma)) \qquad \forall(\Box^k(\Diamond E_\phi \leftarrow \Gamma)),\ \forall(\Box^{k+1}(\phi \leftarrow E_\phi)) \tag{4}$$
$$\forall(\Box^k(\phi \wedge \psi \leftarrow \Gamma)) \qquad \forall(\Box^k(E_{\phi\wedge\psi} \leftarrow \Gamma)),\ \forall(\Box^k(\phi \leftarrow E_{\phi\wedge\psi})),\ \forall(\Box^k(\psi \leftarrow E_{\phi\wedge\psi})) \tag{5}$$
$$\forall(\Box^k((\phi \leftarrow \psi) \leftarrow \Gamma)) \qquad \forall(\Box^k(\phi \leftarrow \psi, \Gamma)) \tag{6}$$
$$\forall(\Box^k(\phi \leftarrow \Gamma, \Box\psi, \Delta)) \qquad \forall(\Box^k(\phi \leftarrow \Gamma, \Box E_\psi, \Delta)),\ \forall(\Box^{k+1}(E_\psi \leftarrow \psi)) \tag{7}$$
$$\forall(\Box^k(\phi \leftarrow \Gamma, \Diamond\psi, \Delta)) \qquad \forall(\Box^k(\phi \leftarrow \Gamma, \Diamond E_\psi, \Delta)),\ \forall(\Box^{k+1}(E_\psi \leftarrow \psi)) \tag{8}$$
$$\forall(\Box^k(\phi \leftarrow \Gamma, \psi \vee \zeta, \Delta)) \qquad \forall(\Box^k(\phi \leftarrow \Gamma, E_{\psi\vee\zeta}, \Delta)), \forall(\Box^k(E_{\psi\vee\zeta} \leftarrow \psi)), \forall(\Box^k(E_{\psi\vee\zeta} \leftarrow \zeta)) \tag{9}$$
$$\forall(\Box^k(\phi \leftarrow \Gamma, \psi \wedge \zeta, \Delta)) \qquad \forall(\Box^k(\phi \leftarrow \Gamma, \psi, \zeta, \Delta)) \tag{10}$$

It is easily seen that P' is an MProlog program, and the translation is done in $O(n^2)$ steps, where n is the size of P''.

Suppose that θ is a correct answer in L for $P' \cup \{G'\}$. Let M be an arbitrary L-model of P. We show that $M \vDash \forall(\xi\theta)$. Let M' be an L-model with the same frame as M such that the content of each world w in M' is the least extension of the content of w in M with the property that for any formula ϕ with E_ϕ occurring in P', $M', w \vDash \forall(E_\phi \leftarrow \phi)$. Thus M' is an L-model of P' and $M' \vDash \forall(E_\xi \rightarrow \xi)$. Since θ is a correct answer in L for $P' \cup \{G'\}$, we have $M' \vDash \forall(E_\xi\theta)$, and hence $M' \vDash \forall(\xi\theta)$. It follows that $M \vDash \forall(\xi\theta)$ (because M' differs from M only on the semantics of new predicate symbols). Therefore every correct answer in L for $P' \cup \{G'\}$ is a correct answer in L for $P \cup \{G\}$.

For the conversion, suppose that θ is a correct answer in L for $P \cup \{G\}$. Let M' be an L-model of P'. Thus M' is also an L-model of P and $M' \vDash \forall(E_\xi \leftarrow \xi)$. Since θ is a correct answer in L for $P \cup \{G\}$, we have $M' \vDash \forall(\xi\theta)$, and hence $M' \vDash \forall(E_\xi\theta)$. Therefore every correct answer in L for $P \cup \{G\}$ is a correct answer in L for $P' \cup \{G'\}$. □

4 Model Generators

Throughout this work, if not stated otherwise, L is one of the logics KD, T, KDB, B, $KD4$, $S4$, $KD5$, $KD45$, $S5$. In this section, we investigate a structure called a *model generator*, which is a set representation of a Kripke model. We show that for every model generator I there exists a model called the *standard L-model* of I which is a least L-model of I.

Definition 4.1 A *model generator* is a set of ground atoms not containing $\Diamond$, $\langle\top\rangle$, $\top$. A model generator I is called an *L-normal* model generator if $L \notin \{KD5, KD45, S5\}$, or $L = KD5$ and I contains only atoms of the form E, ∇E, or $\Box\nabla E$, or $L \in \{KD45, S5\}$ and I contains only atoms of the form E or ∇E.

Given an L-normal model generator I, we can construct a least L-model for it by building an L-model graph realizing I (cf. [29]). Formulae of the form $\Box\alpha$ are realized in the usual way; a formula of the form $\langle E\rangle\alpha$ is realized at a world w by connecting w to a world identified by $w\langle E\rangle$ and adding α to that world. To guarantee the constructed model graph to be the smallest, each new world is connected to an empty world at the time of its creation. Sometimes the accessibility relation is extended to satisfy all of the L-frame restrictions. In this section we want to *define* a least L-model graph for I rather than giving a procedure to construct it. We simulate the process of constructing a least L-model graph for a model generator I by the two following definitions.

Definition 4.2 Let I be a model generator. The *L-extension* of I, denoted by $Ext_L(I)$, is the least (w.r.t. $\subseteq$) model generator J such that $I \subseteq J$ and:

- if $L \in \{T, B, S4\}$ and $\triangle\Box\alpha \in J$ then $\triangle\alpha \in J$;

- if $L \in \{KDB, B\}$ and $\triangle\nabla\Box\alpha \in J$ then $\triangle\alpha \in J$;
- if $L \in \{KD4, S4\}$ and $\triangle\Box\alpha \in J$ then $\triangle\Box\Box\alpha \in J$;
- if $L = KD5$ and $\Box\Box E \in J$ then $\Box E \in J$;
- if $L = S5$ and $\Box E \in J$ then $E \in J$.

Note that $Ext_{KD45}(I) = I$, and if $\triangle\Box\alpha \in Ext_L(I)$ then it is not neccessary that $\triangle\langle E\rangle\alpha \in Ext_L(I)$. It can be shown that for any L-model M, $M \vDash Ext_L(I)$ iff $M \vDash I$.

Definition 4.3 Let I be an L-normal model generator. The *standard L-model* of I is defined as follows. Let J be the set specified by: $J = Ext_L(I) \cup \{\langle\top\rangle\top\}$ for $L \in \{KD45, S5\}$, $J = Ext_L(I) \cup \{\langle\top\rangle\top, \Box\langle\top\rangle\top\}$ for $L = KD5$, and $J = Ext_L(I) \cup \{\Box^k\langle\top\rangle\top \mid k \geq 0\}$ for other cases. Let $W_0 = EdgeLabels^*$ (i.e. the set of finite sequences of elements of $\{\langle E\rangle \mid E \in \mathcal{B} \cup \{\top\}\}$), $\tau = \epsilon$, $H(\tau) = J$. Let $R_0 \subseteq W_0 \times W_0$ and $H(u)$, for $u \in W_0$, $u \neq \tau$, be the least sets such that:

- if $\langle E\rangle\alpha \in H(w)$, then $R_0(w, w\langle E\rangle)$ and $\{E, \alpha\} \subseteq H(w\langle E\rangle)$;
- if $\Box\alpha \in H(w)$ and $R_0(w, w\langle E\rangle)$, then $\alpha \in H(w\langle E\rangle)$.

Let $R \subseteq W_0 \times W_0$ be the least extension of R_0 that satisfies all L-frame restrictions except seriality[6], and let W be W_0 without worlds not reachable directly nor indirectly from τ (w.r.t. R). We call the model graph $\langle W, \tau, R, H\rangle$ the *standard L-model graph* of I, and its corresponding model M the *standard L-model* of I. We call R_0 the *skeleton* of M. By the *standard $\Diamond$-realization function on M* we call the $\Diamond$-realization function σ defined as follows: if $R_0(w, w\langle E\rangle)$ then $\sigma(w, \langle E\rangle) = w\langle E\rangle$, else $\sigma(w, \langle E\rangle)$ is undefined.

Note that a world (being a sequence of labeled modal operators) in a standard L-model has length 0 or 1 for $L \in \{KD45, S5\}$, and has length bounded by 2 for $L = KD5$.

Definition 4.4 An atom $\nabla_1 \ldots \nabla_n E$ is called an *instance* of an atom $\nabla'_1 \ldots \nabla'_n E'$ if there exists a substitution θ such that $E = E'\theta$, and for $1 \leq i \leq n$, $\nabla'_i = \Box$, or $\nabla_i = \Diamond$, or $\nabla_i = \langle F\rangle$ and $\nabla'_i = \langle F'\rangle$ and $F = F'\theta$. A modality $\triangle$ is called an *instance* of $\triangle'$, and we also say that $\triangle'$ is *equal to or more general* than $\triangle$ (hereby we define a *pre-order between modalities*), if $\triangle E$ is an instance of $\triangle' E$ for some ground classical atom E.

We give below an auxiliary lemma.

Lemma 4.1 *Let I be an L-normal model generator and $M = \langle W, \tau, R, H\rangle$ the standard L-model graph of I. Let $w = \langle E_1\rangle \ldots \langle E_k\rangle$ be a world of M and $\triangle = w$ be a modality. Then for α not containing $\top$, $\alpha \in H(w)$ iff $\triangle\alpha$ is an instance of some atom from $Ext_L(I)$.*

This lemma can be proved by induction on the length of w in a straightforward way.

The following lemma states that Definition 4.3 is admissible (i.e. the standard L-model of I is really an L-model of I).

Lemma 4.2 *Let I be an L-normal model generator, M the standard L-model of I, and σ the standard $\Diamond$-realization function on M. Then M is an L-model and $M, \sigma \vDash I$.*

Proof. Clearly, M is an L-model. Let R_0 be the skeleton of M. We prove by induction on the length of α that, for any $w \in W$, if $\alpha \in H(w)$ then $M, \sigma, w \vDash \alpha$. Let $\triangle = w = \langle E_1\rangle \ldots \langle E_k\rangle$.

The cases when α is a classical atom or $\alpha = \langle E\rangle\beta$ are trivial.

Suppose that $\alpha = \Box\beta$. Let u be a world such that $R(w, u)$ holds. We show that $\beta \in H(u)$. There are the following cases to consider:

- Case $u = w\langle E\rangle$ and $R_0(w, w\langle E\rangle)$: The assertion holds by the definition of M.

[6]Note that the condition about seriality is guaranteed by the set $(J - Ext_L(I))$, and the other L-frame restrictions are Horn clauses.

- Case $u = w$ and $L \in \{T, B, S4\}$: Since $\Box\beta \in H(w)$, by Lemma 4.1, $\triangle\Box\beta$ is an instance of some atom from $Ext_L(I)$. By the definition of $Ext_L(I)$, it follows that $\triangle\beta$ is an instance of some atom from $Ext_L(I)$. Hence $\beta \in H(u)$.

- Case $w = u\langle E_k\rangle$ and $L \in \{KDB, B\}$: Since $\Box\beta \in H(u\langle E_k\rangle)$, either $\Box\Box\beta$ or $\langle E_k\rangle\Box\beta$ belongs to $H(u)$. By Lemma 4.1, $\langle E_1\rangle \ldots \langle E_{k-1}\rangle\Diamond\Box\beta$ is an instance of some atom from $Ext_L(I)$. By the definition of $Ext_L(I)$, it follows that $\langle E_1\rangle \ldots \langle E_{k-1}\rangle\beta$ is an instance of some atom from $Ext_L(I)$. Hence $\beta \in H(u)$.

- Case $R_0^h(w, u)$, $h \geq 1$, and $L \in \{KD4, S4\}$: Since $\Box\beta \in H(w)$, by Lemma 4.1, $\triangle\Box\beta$ is an instance of some atom from $Ext_L(I)$. By the definition of $Ext_L(I)$, it follows that $\triangle\Box^h\beta$ is an instance of some atom from $Ext_L(I)$. Hence $\beta \in H(u)$.

- Case $L = KD5$, $w = \langle E_1\rangle$, and ($u = \langle E\rangle$ or $u = \langle E\rangle\langle F\rangle$) for some E, F: Since $\Box\beta \in H(w)$, we have $\Box\Box\beta \in H(\tau)$. By the definition of Ext_L, we have $\Box\beta \in H(\tau)$. Hence $\beta \in H(u)$.

- Case $L = S5$ and $w = u = \tau$: This case is trivial.

Note that the case $L = KD45$ and other cases for $L \in \{KD5, S5\}$ are included in the first case of the list. Since $R(w, u)$ holds, the above list contains all of the cases that can occur. □

Here is the main theorem of this section:

Theorem 4.3 *Let I be an L-normal model generator. Then the standard L-model of I is a least L-model of I.*

Proof. Let $M = \langle W, \tau, R, H\rangle$ be the standard L-model graph of I, σ the standard $\Diamond$-realization function and R_0 the skeleton of the standard L-model of I. Let $N = \langle D, W', \tau', R', m\rangle$ be an arbitrary L-model of I and σ' a $\Diamond$-realization function on N such that $N, \sigma' \vDash I \cup \bigcup_{k=0}^{\omega}\{\Box^k\langle\top\rangle\top\}$ (such a σ' exists because N is a serial model of I).

In the following, $w, u \in W$, $w', u' \in W'$, and E, F are arbitrary.

Define $r \subseteq W \times W'$ to be the least relation such that:

- $r(\tau, \tau')$;

- if $r(w, w')$ and $R_0(w, w\langle E\rangle)$ hold, and $\sigma'(w', \langle E\rangle)$ is defined, then $r(w\langle E\rangle, \sigma'(w', \langle E\rangle))$;

- if $L \notin \{KD5, KD45, S5\}$, $r(w, w')$ and $R'(w', u')$ hold, then $r(w\langle\top\rangle, u')$;

- if $L \in \{KD5, KD45, S5\}$ and $R'(\tau', u')$ holds, then $r(\langle\top\rangle, u')$;

- if $L = KD5$, and $R'(\tau', w')$ and $R'(w', u')$ hold for some w', then $r(\langle\top\rangle\langle\top\rangle, u')$.

We first show that if $r(w, w')$ and $R'(w', u')$ hold then there exists u such that $r(u, u')$ and $R(w, u)$ hold. For $L \notin \{KD5, KD45, S5\}$, choose $u = w\langle\top\rangle$. For $L \in \{KD45, S5\}$, choose $u = \langle\top\rangle$. For the case $L = KD5$ and $w = \tau$, choose $u = \langle\top\rangle$. For the case $L = KD5$ and $w \neq \tau$, choose $u = \langle\top\rangle\langle\top\rangle$. It is easy to verify that the assertion holds for the chosen u.

We now show that if $r(u, u')$ and $\alpha \in H(u)$, then $N, \sigma', u' \vDash \alpha$. We prove this by induction on the length of u. Suppose that $r(u, u')$ holds and $\alpha \in H(u)$. The case $u = \tau$ is trivial. Let $u = w\langle E\rangle$ and inductively assume that the assertion holds when u is replaced by w. There are the following cases:

- $u' = \sigma'(w', \langle E\rangle)$ (and $\sigma'(w', \langle E\rangle)$ is defined), $r(w, w')$, and $R_0(w, w\langle E\rangle)$, for some w';

- $E = \top$, $r(w, w')$, and $R'(w', u')$, for some w';

- $u = \langle\top\rangle$, $L \in \{KD5, KD45, S5\}$, and $R'(\tau', u')$;

- $u = \langle\top\rangle\langle\top\rangle$, $L = KD5$, and $R'(\tau', w')$ and $R'(w', u')$ hold for some w'.

Consider the first case. Since $\alpha \in H(u)$, either $\Box\alpha \in H(w)$ or $\langle E\rangle\alpha \in H(w)$. By the inductive assumption, either $N, \sigma', w' \vDash \Box\alpha$ or $N, \sigma', w' \vDash \langle E\rangle\alpha$. Hence, $N, \sigma', \sigma'(w', \langle E\rangle) \vDash \alpha$, which means that $N, \sigma', u' \vDash \alpha$. Consider the second case. Since $\alpha \in H(u)$, it follows that $\Box\alpha \in H(w)$. By the inductive assumption, $N, \sigma', w' \vDash \Box\alpha$, and hence $N, \sigma', u' \vDash \alpha$ since $R'(w', u')$. Consider the third case. Since $\alpha \in H(u)$, it follows that $\Box\alpha \in H(\tau)$. By the inductive assumption, $N, \sigma', \tau' \vDash \Box\alpha$, and hence $N, \sigma', u' \vDash \alpha$ since $R'(\tau', u')$. Consider the last case. Since $\alpha \in H(u)$, it follows that $\Box\Box\alpha \in H(\tau)$. By the inductive assumption, $N, \sigma', \tau' \vDash \Box\Box\alpha$, and hence $N, \sigma', u' \vDash \alpha$ since $R'(\tau', w')$ and $R'(w', u')$.

We show that if $r(w, w')$ and $R_0(w, w\langle E\rangle)$ hold, then $\sigma'(w', \langle E\rangle)$ is defined. The case $E = \top$ is trivial. Suppose that $r(w, w')$ and $R_0(w, w\langle E\rangle)$ hold, and $E \neq \top$. There exists $\langle E\rangle\alpha \in H(w)$ for some α. By the above proved assertion, $N, \sigma', w' \vDash \langle E\rangle\alpha$. Hence $\sigma'(w', \langle E\rangle)$ is defined. Therefore, the second condition in the definition of r can be simplified to "if $r(w, w')$ and $R_0(w, w\langle E\rangle)$ hold, then $r(w\langle E\rangle, \sigma'(w', \langle E\rangle))$".

To prove $M \leq_r N$, it remains to show that if $r(w, w')$ and $R(w, u)$ hold, then there exists $u' \in W'$ such that $r(u, u')$ and $R'(w', u')$ hold. Suppose that $r(w, w')$ and $R(w, u)$ hold. There are the following cases to consider:

- Case $u = w\langle E\rangle$: Choose $u' = \sigma'(w', \langle E\rangle)$.
- Case $u = w$ and $L \in \{T, B, S4\}$: Choose $u' = w'$.
- Case $w = u\langle E\rangle$ and $L \in \{KDB, B\}$: It is straightforward to prove by induction on the length of x that if $r(x, x')$ holds then either $x = \tau$ and $x' = \tau'$ or $x = y\langle F\rangle$ for some F and there exists y' such that $R'(y', x')$ and $r(y, y')$ hold. Hence, there exists u' such that $R'(u', w')$ and $r(u, u')$ hold. For such chosen u' we have $R'(w', u')$ (due to symmetry).
- Case $u = w\langle E_1\rangle \ldots \langle E_k\rangle$ and $L \in \{KD4, S4\}$: Let $u'_0 = w'$ and $u'_i = \sigma'(u'_{i-1}, \langle E_i\rangle)$ for $1 \leq i \leq k$. Choose $u' = u'_k$.
- Case $u = \tau$ and $L = S5$: Choose $u' = \tau'$.
- Case $u = \langle E\rangle$ and $L \in \{KD5, KD45, S5\}$: Choose $u' = \sigma'(\tau', \langle E\rangle)$.
- Case $u = \langle E\rangle\langle F\rangle$ and $L = KD5$: Choose $u' = \sigma'(\sigma'(\tau', \langle E\rangle), \langle F\rangle)$.

Since $R(w, u)$ holds, the above list contains all of the cases that can occur. It is straightforward to check that $R'(w', u')$ and $r(u, u')$ hold. This completes the proof. □

5 A Fixpoint Semantics for Modal Logic Programs

In this section, we show that for every MProlog program P there exists the least (w.r.t. $\subseteq$) model generator $I_{L,P}$ such that $P \vDash_L I_{L,P}$ and the standard L-model of $I_{L,P}$ is a least L-model of P. The model generator $I_{L,P}$ is the least fixpoint of a certain operator.

Given an MProlog program P and an L-normal model generator I, we want to apply the clauses of P to I to obtain another model generator and repeat the process for the new model generator until obtaining a fixpoint. There are some questions related with this aim:

- What will be applied? We can use not only the program clauses of P but also their consequences. For example, given a program clause $\Box(A \leftarrow B_1, \ldots, B_n)$ in the logic T, we can use also the clause $A \leftarrow B_1, \ldots, B_n$.
- What will the program clauses be applied to? They are the atoms of I and consequences of I. For example, suppose that P contains $E \leftarrow \Diamond F$, $I = \{F\}$, and the base logic is T. We can apply the program clause to $\Diamond F$, which is a consequence of F in the logic T.
- How will the program clauses be applied? Let us give an example. Suppose that P contains $\Box(E \leftarrow F_1, F_2, F_3)$, I contains $\Box F_1$, $\langle F\rangle F_2$ and $\langle F\rangle F_3$, and the base logic is KD. The result of the application of the rule to the mentioned atoms is $\langle F\rangle E$.

- What should be done if the resulting model generator is not in the L-normal form? The answer is simple: normalize it.

We will give adequate answers for the above questions. As for the first one, *L-instances* (defined below) of the program clauses of P are used as rules to be applied.

Definition 5.1 A program clause $\Box^k(A \leftarrow B_1, \ldots, B_n)$ is called an *L-instance* of a program clause $\Box^h(A' \leftarrow B'_1, \ldots, B'_n)$ if there exists a substitution θ such that $(A \leftarrow B_1, \ldots, B_n) = (A' \leftarrow B'_1, \ldots, B'_n)\theta$ and:

- case $L = KD : k = h$,
- case $L = T : k \leq h$,
- case $L = KDB : k = h - 2l$, for some $l \geq 0$,
- case $L = B : k \leq h$,
- case $L = KD4 : k \geq h > 0$ or $k = h = 0$,
- case $L = S4 : k = 0$ or ($k > 0$ and $h > 0$),
- case $L = KD5 : k = h \in \{0, 1, 2\}$ or ($k = 1$ and $h = 2$),
- case $L = KD45 : k = h \in \{0, 1\}$,
- case $L = S5 : k = 0$ or $k = h = 1$.

It can be shown that if a program clause ϕ is an L-instance of ϕ', then $\forall(\phi' \rightarrow \phi)$ is L-valid. L-instances of program clauses are applied to atoms of the *L-saturation* (defined in the following) of a given model generator.

Definition 5.2 Let I be an L-normal model generator. The *L-saturation* of I, denoted by $Sat_L(I)$, is the least set J of ground atoms such that $I \subseteq J$ and:

- if $L \in \{T, B, S4\}$:
  - if $\triangle\Box\alpha \in J$ then $\triangle\alpha \in J$,
  - if $\triangle E \in J$ and $\triangle$ does not contain $\Diamond$, then $\triangle\Diamond E \in J$;
- if $L \in \{KDB, B\}$:
  - if $\triangle\nabla\Box\alpha \in J$ then $\triangle\alpha \in J$,
  - if $\triangle E \in J$ and $\triangle$ does not contain $\Diamond$, then $\triangle\Box\Diamond E \in J$;
- if $L \in \{KD4, S4\}$:
  - if $\triangle\Box\alpha \in J$ then $\triangle\Box\Box\alpha \in J$,
  - if $\triangle\nabla_1\nabla_2 E \in J$ then $\triangle\Diamond E \in J$;
- if $L = KD5$:
  - if $\nabla E \in J$ then $\Box\Diamond E \in J$ and $\Box\Box\Diamond E \in J$,
  - if $\Box\Box E \in J$ then $\Box E \in J$ and $\Box\Box\Box E \in J$,
  - if $\Box\langle F\rangle E \in J$ then $\Box\Box\Diamond E \in J$;
- if $L = KD45$:
  - if $\nabla E \in J$ then $\Box\Diamond E \in J$,
  - if $\Box E \in J$ then $\Box\Box E \in J$;
- if $L = S5$:

– the conditions as for the case $L = KD45$, plus
– if $\Box E \in J$ then $E \in J$,
– if $E \in J$ then $\Diamond E \in J$.

Note that $Sat_L(I)$ is a superset of $Ext_L(I)$.
It can be shown that for any L-model M, $M \models Sat_L(I)$ iff $M \models I$.

Definition 5.3 The *forward labeled form* of an atom α is the atom α' such that if α is of the form $\triangle\Diamond E$ then $\alpha' = \triangle\langle E\rangle E$, else $\alpha' = \alpha$.

We can "directly" apply the rules of a program using the operator $T_{^0L,P}$ defined in the following.

Definition 5.4 Let P be an MProlog program. The operator $T_{^0L,P}$ is defined as follows: for $I = Sat_L(J)$ with J being an L-normal model generator, $T_{^0L,P}(I)$ is the least (w.r.t. $\subseteq$) model generator such that if $\Box^k(A \leftarrow B_1, \ldots, B_n)$ is a ground L-instance of some program clause of P, and $\triangle$ is a maximally general[7] ground modality with length k, not containing $\Diamond$, such that for every $1 \leq i \leq n$, $\triangle B_i$ is an instance of some atom from I, then the forward labeled form of $\triangle A$ belongs to $T_{^0L,P}(I)$.

For an L-normal model generator I, $T_{^0L,P}(Sat_L(I))$ is a model generator, but not necessary in the L-normal form (for $L \in \{KD5, KD45, S5\}$). We can normalize it as described below.

Definition 5.5 Let $L \in \{KD5, KD45, S5\}$, P be an MProlog program and $I = T_{^0L,P}(Sat_L(I'))$ for some L-normal model generator I'. We define the *L-normal form* of I, denoted by $NF_L(I)$, to be the least set J such that:

- Case $L \in \{KD45, S5\}$: J contains all atoms of the form E or ∇E of I, and if $\nabla\nabla' E \in I$ (note that $\nabla' = \Box$ or $\nabla' = \langle E\rangle$) then $\nabla' E \in J$.
- Case $L = KD5$: J contains all atoms of the form E, ∇E, or $\Box\nabla E$ of I, and if $\Box\nabla\nabla' E \in I$ or $\langle F\rangle\nabla' E \in I$ (note that $\nabla' = \Box$ or $\nabla' = \langle E\rangle$) then $\Box\nabla' E \in J$.

We are now at the position to define the "direct consequences" operator $T_{L,P}$, which has the least fixpoint being the least *L-model generator of P* (see Definition 5.7).

Definition 5.6 Let P be an MProlog program. The operator $T_{L,P}$ is defined as follows: if $L \notin \{KD5, KD45, S5\}$ then $T_{L,P}(I) = T_{^0L,P}(Sat_L(I))$, else $T_{L,P}(I) = NF_L(T_{^0L,P}(Sat_L(I)))$.

Denote $T_{L,P} \uparrow 0 = \emptyset$, and $T_{L,P} \uparrow n = T_{L,P}(T_{L,P} \uparrow (n-1))$ for $n > 0$.
We give here a digression about fixpoints. Let $T : \mathcal{P}(U) \rightarrow \mathcal{P}(U)$ be an operator that maps each subset of U to a subset of U. T is said to be *monotonic* if for every subsets V and V' of U, if $V \subseteq V'$ then $T(V) \subseteq T(V')$. T is said to be *continuous* if for every set $\mathcal{V}$ of subsets of U, $T(\bigcup \mathcal{V}) = \bigcup\{T(V) \,|\, V \in \mathcal{V}\}$. T is said to be *compact* if for every subset V of U and for every $a \in T(V)$, there exists a finite subset V' of V such that $a \in T(V')$. It is known that if T is monotonic and compact then T is continuous. A set V is a *fixpoint* of T if $T(V) = V$. The fixpoint theorem by Knaster and Tarski says that if T is monotonic and continuous, then T has the least fixpoint specified by $\bigcup_{n=0}^{\omega} T^n(\emptyset)$.

Lemma 5.1 *For P being an MProlog program, the operator $T_{L,P}$ is monotonic and continuous, and it has the least fixpoint $T_{L,P} \uparrow \omega = \bigcup_{n=0}^{\omega} T_{L,P} \uparrow n$.*

The operators Sat_L, $T_{^0L,P}$, and NF_L are all increasingly monotonic and compact, and so is $T_{L,P}$. It follows that $T_{L,P}$ is continuous and has the least fixpoint as above specified.

Notation 5.1 *Denote the least fixpoint of $T_{L,P}$ by $I_{L,P}$, and the standard L-model of $I_{L,P}$ by $M_{L,P}$.*

Lemma 5.2 *Let P be an MProlog program. Then $P \models_L I_{L,P}$*

[7] w.r.t. the pre-order between modalities specified in Definition 4.4

Proof. Let $M = \langle D, W, \tau, R, m\rangle$ be an L-model of P. Let σ be the $\Diamond$-realization function defined as follows: if $R(w,u)$ holds and $M, u \vDash E$, then $\sigma(w, \langle E\rangle) = u$ for such a u. It is straightforward to prove by induction on n that $M, \sigma \vDash T_{L,P} \uparrow n$. In fact, if $M, \sigma \vDash T_{L,P} \uparrow n$, then $M, \sigma \vDash Sat_L(T_{L,P} \uparrow n)$, $M, \sigma \vDash T_{0L,P}(Sat_L(T_{L,P} \uparrow n))$, and for $L \in \{KD5, KD45, S5\}$ $M, \sigma \vDash NF_L(T_{0L,P}(Sat_L(T_{L,P} \uparrow n)))$. Therefore $M, \sigma \vDash I_{L,P}$ what proves the lemma. □

Definition 5.7 Let P be an MProlog program. An L-normal model generator I is called an *L-model generator of P* if $T_{L,P}(I) \subseteq I$.

As a property of the least fixpoint, $I_{L,P}$ is the least (w.r.t. $\subseteq$) L-model generator of P.

Lemma 5.3 *Let P be an MProlog program, I an L-normal model generator of P, and M the standard L-model of I. Then M is an L-model of P.*

Proof. Let $\langle W, \tau, R, H\rangle$ be the standard L-model graph of I and σ the standard $\Diamond$-realization function on M. It is sufficient to prove that for any ground L-instance $\Box^k(A \leftarrow B_1, \ldots, B_n)$ of some program clause of P, for any $w \in W$ with $|w| = k$ (note that $w \in EdgeLabels^*$), $M, w \vDash (A \leftarrow B_1, \ldots, B_n)$. Suppose that $M, w \vDash B_i$ for all $1 \le i \le n$. We show that $M, w \vDash A$.

Let $\triangle = w = \langle E_1\rangle \ldots \langle E_k\rangle$. We first show that for any ground simple atom B of the form E, $\Box E$, or $\Diamond E$, if $M, w \vDash B$ then $\triangle B$ is an instance of some atom from $Sat_L(I)$. If B is of the form E or $\Box E$, then by Lemma 4.1, $\triangle B$ is an instance of some atom from $Sat_L(I)$ (since $Ext_L(I) \subseteq Sat_L(I)$). Now consider the case when B is of the form $\Diamond E$. There exists u such that $R(w,u)$ and $M, u \vDash E$. There are the following cases to consider:

- Case $u = w\langle E_{k+1}\rangle$ for some $E_{k+1} \in \mathcal{B} \cup \{\top\}$: By Lemma 4.1, $\triangle\langle E_{k+1}\rangle E$ is an instance of some atom from $Sat_L(I)$, hence so is $\triangle\Diamond E$.
- Case $u = w$ and $L \in \{T, B, S4\}$: By Lemma 4.1, $\triangle E$ is an instance of some $\triangle' E \in Sat_L(I)$. Hence $\triangle\Diamond E$ is an instance of $\triangle'\Diamond E$, which belongs to $Sat_L(I)$ by definition.
- Case $u = \langle E_1\rangle \ldots \langle E_{k-1}\rangle$ and $L \in \{KDB, B\}$: By Lemma 4.1, $\langle E_1\rangle \ldots \langle E_{k-1}\rangle E$ is an instance of some $\triangle' E \in Sat_L(I)$. Hence $\triangle\Diamond E$ is an instance of $\triangle'\Box\Diamond E$, which belongs to $Sat_L(I)$ by definition.
- Case $u = w\langle E_{k+1}\rangle \ldots \langle E_{k+j}\rangle$ for some $E_{k+1}, \ldots, E_{k+j} \in \mathcal{B} \cup \{\top\}$, and $L \in \{KD4, S4\}$: By Lemma 4.1, $\triangle\langle E_{k+1}\rangle \ldots \langle E_{k+j}\rangle E$ is an instance of some $\triangle' E \in Sat_L(I)$. Let $\triangle' = \triangle''\nabla_1 \ldots \nabla_j$. Thus $\triangle\Diamond E$ is an instance of $\triangle''\Diamond E$, which belongs to $Sat_L(I)$ by definition.
- Case $u = \tau$ and $L = S5$: We have $E \in Sat_L(I)$. Hence, by the definition of Sat_L, $\{\Diamond E, \Box\Diamond E\} \subseteq Sat_L(I)$. Therefore $\triangle\Diamond E$ is an instance of some atom from $Sat_L(I)$.
- Case $u = \langle F\rangle$ for some F, $w \neq \tau$, and $L \in \{KD5, KD45, S5\}$: By Lemma 4.1, $\langle F\rangle E$ is an instance of some $\nabla E \in Sat_L(I)$. By the definition of Sat_L, $\Box\Diamond E \in Sat_L(I)$, and we also have $\Box\Box\Diamond E \in Sat_L(I)$ if $L = KD5$. Hence $\triangle\Diamond E$ is an instance of some atom from $Sat_L(I)$.
- Case $u = \langle F_1\rangle\langle F_2\rangle$ for some F_1 and F_2, $w \neq \tau$, and $L = KD5$: By Lemma 4.1, $\langle F_1\rangle\langle F_2\rangle E$ is an instance of some $\triangle' E \in Sat_L(I)$. From the definition of Sat_L, we claim that $\Box\Diamond E$ and $\Box\Box\Diamond E$ are instances of some atoms from $Sat_L(I)$, hence so is $\triangle\Diamond E$.

Since $R(w,u)$ holds, the above list contains all of the cases that can occur.

By Lemma 4.2, $M, \sigma \vDash I$. Since $M, w \vDash B_i$ for $1 \le i \le n$, it follows that $\triangle B_i$ is an instance of some atom from $Sat_L(I)$. Consequently, $\triangle A$ is an instance of some atom α from $T_{0L,P}(Sat_L(I))$. For $L \notin \{KD5, KD45, S5\}$, we have $T_{0L,P}(Sat_L(I)) = T_{L,P}(I) \subseteq I$, hence $M, \sigma \vDash \alpha$ and $M, w \vDash A$. Suppose that $L \in \{KD5, KD45, S5\}$. Let $\alpha = \triangle' A'$, where A' is the forward labeled form of A. Since $M, \sigma \vDash I$ and $\alpha \in T_{0L,P}(Sat_L(I))$, we have $M, \sigma \vDash \triangle'\top$. Since $T_{L,P}(I) \subseteq I$, we have $M, \sigma \vDash NF_L(\{\alpha\})$. These together imply that $M, \sigma \vDash \triangle' A$. Since $\triangle$ is an instance of $\triangle'$, we conclude that $M, w \vDash A$. □

Here is the main theorem of this section:

Theorem 5.4 *For an MProlog program P, $M_{L,P}$ is a least L-model of P.*

Proof. By Lemma 5.3, $M_{L,P}$ is an L-model of P. Let M be any L-model of P. By Lemma 5.2, $M \models I_{L,P}$. Hence, by Theorem 4.3, $M_{L,P} \leq M$. Therefore $M_{L,P}$ is a least L-model of P. □

6 An SLD-Resolution Calculus for Modal Logic Programs

The fixpoint semantics can be viewed as a bottom-up method for computing answers. It repeatedly applies clauses of a given program P in order to compute the set $I_{L,P}$ of facts derivable in L from the program. Given an atom α from $I_{L,P}$, the process of tracing back the derivation of α in L from P is called top-down because it reduces the atom, treated as a goal, to subgoals. A more general problem is finding correct answers for an MProlog goal with respect to an MProlog program. This problem is reduced to the previous by unification and a lifting technique.

In this section, we provide a calculus called *SLD-resolution* for solving the problem of finding correct answers for $P \cup \{G\}$, where P is an MProlog program and G is an MProlog goal. The main work to do here is to specify a reverse of the operator $T_{L,P}$. While $T_{L,P}$ acts on model generators (with only ground atoms), the expected reverse of $T_{L,P}$ will act on goals (with variables). The operator $T_{L,P}$ is a composition of Sat_L, $T_{0L,P}$, and NF_L (for $L \in \{KD5, KD45, S5\}$). So, we have to investigate reversion of these operators. We first define (general) goals, which may contain labeled modal operators.

Definition 6.1 A *goal atom* is an atom of the form $\triangle E$ or $\triangle\Diamond E$, where $\triangle$ is a modality without $\langle\top\rangle$ and $\Diamond$. A *goal* is a clause of the form $\leftarrow \alpha_1, \ldots, \alpha_k$, where each α_i is a goal atom.

In the definition of $T_{0L,P}$, instantiating and forward labeling are used. For example, applying the clause $\Diamond E \leftarrow \Diamond F$ to $\Box F$ we obtain $\langle E\rangle E$; here $\Diamond F$ is an instance of $\Box F$, and $\langle E\rangle E$ is the forward labeled form of $\Diamond E$. For the reverse of $T_{0L,P}$, we make these tasks explicit.

Definition 6.2 The *backward labeling operator* is the one that converts a goal atom $\triangle\Diamond E$ to $\triangle\langle X\rangle E$, where X is a fresh atom variable. We call $\triangle\langle X\rangle E$ a *backward labeled form* of $\triangle\Diamond E$. If an atom α is obtainable from α' by replacing some modal operators by $\Box$, then we call α a *$\Box$-lifting form* of α' and the operator that converts α' to α the *$\Box$-lifting operator*.

We now define the reverse of $T_{0L,P}$. It is an operator that derives a new goal from a goal and a program clause.

Definition 6.3 Let $G = \leftarrow \alpha_1, \ldots, \alpha_i, \ldots, \alpha_k$ be a goal and $\phi = \Box^s(A \leftarrow B_1, \ldots, B_m)$ a program clause. Then G' is *derived* from G and ϕ in L using mgu θ if the following conditions hold:

- $\alpha_i = \triangle A'$, where $\triangle$ is a modality not containing $\Diamond$, is an atom called the *selected* atom.
- $|\triangle| = h$ and $\Box^h(A \leftarrow B_1, \ldots, B_m)$ is an L-instance of ϕ.
- If $L = KD5$ and $|\triangle| = 2$ then $\triangle$ is of the form $\Box\nabla$.
- θ is an mgu of A' and the forward labeled form of A.
- G' is the goal $\leftarrow (\alpha_1, \ldots, \alpha_{i-1}, \triangle B_1, \ldots, \triangle B_m, \alpha_{i+1}, \ldots, \alpha_k)\theta$.

In resolution terminology, G' is called an *L-resolvent* of G and ϕ.

In the bottom-up context we deal with model generators. Given a model generator we can extend it by applying the saturation operator Sat_L. When an atom of some specific form belongs to the model generator, Sat_L adds another one to it. In the top-down context, goals are objects to deal with, and the reverse of Sat_L has the following form: a goal atom can be replaced by another one.

Definition 6.4 The reverse of Sat_L, denoted by $rSat_L$, is defined as follows. For a goal atom α, $rSat_L(\alpha)$ can be α or

- case $L \in \{T, B, S4\}$:
  - $\triangle\Box\beta$ if $\alpha = \triangle\beta$;
  - $\triangle E$ if $\alpha = \triangle\Diamond E$;
- case $L \in \{KDB, B\}$:
  - $\triangle\langle X\rangle\Box\beta$ if $\alpha = \triangle\beta$;
  - $\triangle E$ if $\alpha = \triangle\Box\Diamond E$;
- case $L \in \{KD4, S4\}$:
  - $\triangle\Box\beta$ if $\alpha = \triangle\Box\Box\beta$;
  - $\triangle\langle X\rangle\Diamond E$ if $\alpha = \triangle\Diamond E$;
- case $L = KD5$:
  - $\langle X\rangle E$ if $\alpha = \Box\Diamond E$ or $\alpha = \Box\Box\Diamond E$;
  - $\Box\Box E$ if $\alpha = \Box E$ or $\alpha = \Box\Box\Box E$;
  - $\Box\langle X\rangle E$ if $\alpha = \Box\Box\Diamond E$;
- case $L = KD45$:
  - $\langle X\rangle E$ if $\alpha = \Box\Diamond E$;
  - $\Box E$ if $\alpha = \Box\Box E$;
- case $L = S5$:
  - as for the case $L = KD45$, or
  - $\Box E$ if $\alpha = E$;
  - E if $\alpha = \Diamond E$;

where X and Y are fresh atom variables.

The operator $rSat_L$ is thus nondeterministic.
The reverse form of NF_L is defined in a similar way.

Definition 6.5 For $L \in \{KD5, KD45, S5\}$, the reverse form of NF_L, denoted by rNF_L, is defined as follows. For a goal atom α, $rNF_L(\alpha)$ can be α or

- $\langle X\rangle\nabla E$ if $L \in \{KD45, S5\}$ and $\alpha = \nabla E$;
- $\langle X\rangle\nabla E$ or $\Box\langle X\rangle\nabla E$ if $L = KD5$ and $\alpha = \Box\nabla E$;

where X is a fresh atom variable.

The operator rNF_L is also nondeterministic. For convenience, if $L \notin \{KD5, KD45, S5\}$ then we assume that $rNF_L(\alpha) = \alpha$. Another solution is assuming $L \in \{KD5, KD45, S5\}$ whenever rNF_L is considered.

We now define SLD-derivation and SLD-refutation.

Definition 6.6 Let $G = \leftarrow \alpha_1, \ldots, \alpha_i, \ldots, \alpha_k$ and $G' = \leftarrow \alpha_1, \ldots, \alpha_{i-1}, \alpha'_i, \alpha_{i+1}, \ldots, \alpha_k$ be goals. We say that G' is *derived* from G using $rSat_L$ (resp. rNF_L) if $\alpha'_i = rSat_L(\alpha_i)$ (resp. $\alpha'_i = rNF_L(\alpha_i)$) in nondeterministic meaning[8]. G' is called a *backward labeled form* of G if α'_i is a backward labeled form of α_i, and called a $\Box$*-lifting form* of G if α'_i is a $\Box$-lifting form of α_i. The atom α_i is called the *selected* atom.

[8]This means that α'_i is one of the possible values of $rSat_L(\alpha_i)$ (resp. $rNF_L(\alpha_i)$).

Definition 6.7 Let P be an MProlog program and G an MProlog goal. An *SLD-derivation* of $P \cup \{G\}$ in L consists of a (finite or infinite) sequence $G_0 = G, G_1, \ldots$ of goals, a sequence $\phi_{i_1}, \phi_{i_2}, \ldots$ of variants of program clauses of P and a sequence $\theta_{i_1}, \theta_{i_2}, \ldots$ of mgu's such that for $i \in \{i_1, i_2, \ldots\}$, G_i is derived from G_{i-1} and ϕ_i in L using θ_i, and for $i \notin \{i_1, i_2, \ldots\}$, G_i is either a backward labeled form or a $\Box$-lifting form of G_{i-1}, or is derived from G_{i-1} using $rSat_L$ or rNF_L.

Each ϕ_{i_j} is a suitable variant of the corresponding program clause so that ϕ_{i_j} does not have any variable which already appears in the derivation up to G_{i_j-1}. This can be achieved by subscripting variables in G by 0 and variables in ϕ_i by i. This process of renaming variables is usually called *standardizing* the variables *apart* (see [24]). Each program clause variant $\phi_{i_1}, \phi_{i_2}, \ldots$ is called an *input clause* of the derivation.

Definition 6.8 An *SLD-refutation* (*refutation* in short) of $P \cup \{G\}$ in L is a finite SLD-derivation of $P \cup \{G\}$ in L which has the empty clause $\diamond$ as the last goal in the derivation. If $G_n = \diamond$, we say the refutation has *length* n.

Definition 6.9 Let P be an MProlog program and G a goal. A *computed answer* θ in L for $P \cup \{G\}$ is the substitution obtained by restricting the composition $\theta_{i_1} \ldots \theta_{i_h}$ to the variables of G, where $\theta_{i_1}, \ldots, \theta_{i_h}$ is the sequence of mgu's used in an SLD-refutation of $P \cup \{G\}$ in L.

Example 6.1 Let G be the goal $\leftarrow \Box p_5(x)$ and P the program that consists of the following clauses

$$\begin{aligned}
&\phi_1 = \Diamond p_1(a) \leftarrow \\
&\phi_2 = \Box(\, p_2(x) \leftarrow p_1(x)\,) \\
&\phi_3 = \Box(\, \Box p_3(x) \leftarrow p_1(x), p_2(x)\,) \\
&\phi_4 = \Box p_4(x) \leftarrow p_3(x) \\
&\phi_5 = \Box(\, p_5(x) \leftarrow p_4(x)\,)
\end{aligned}$$

Here is an SLD-refutation of $P \cup \{G\}$ in KDB with computed answer $\{x/a\}$:

Goals	Input Clauses / Operators	Mgu's
$\leftarrow \Box p_5(x)$		
$\leftarrow \Box p_4(x_1)$	ϕ_5	$\{x/x_1\}$
$\leftarrow p_3(x_2)$	ϕ_4	$\{x_1/x_2\}$
$\leftarrow \langle X\rangle \Box p_3(x_2)$	$rSat_{KDB}$	
$\leftarrow \langle X\rangle p_1(x_4), \langle X\rangle p_2(x_4)$	ϕ_3	$\{x_2/x_4\}$
$\leftarrow \langle p_1(a)\rangle\, p_2(a)$	ϕ_1	$\{x_4/a, X/p_1(a)\}$
$\leftarrow \langle p_1(a)\rangle\, p_1(a)$	ϕ_2	$\{x_5/a\}$
$\diamond$	ϕ_1	ε

Example 6.2 Let $\Box\phi$ stand for "we believe in ϕ". Since $\Diamond\phi = \neg\Box\neg\phi$, the formula $\Diamond\phi$ means "we do not believe that ϕ is false". It can be also interpreted as "it is possible that ϕ holds". Let P be the program containing the following clauses:

$$\begin{aligned}
&\phi_1 = \Box(\, likes_football(husband(x)) \leftarrow woman(x), likes_football(x)\,) \\
&\phi_2 = \Diamond likes_football(x) \leftarrow man(x) \\
&\phi_3 = man(husband(x)) \leftarrow woman(x) \\
&\phi_4 = \Box woman(x) \leftarrow woman(x) \\
&\phi_5 = \Box man(x) \leftarrow man(x) \\
&\phi_6 = \Box likes_football(x) \leftarrow likes_football(x) \\
&\phi_7 = man(Tom) \leftarrow \\
&\phi_8 = woman(Jane) \leftarrow \\
&\phi_9 = woman(Mary) \leftarrow \\
&\phi_{10} = likes_football(Jane) \leftarrow
\end{aligned}$$

Let G be the goal $\leftarrow \Box likes\_football(x)$ and G' be $\leftarrow \Diamond likes\_football(x)$. There are two computed answers in KD for $P \cup \{G\}$: $\{x/husband(Jane)\}$ and $\{x/Jane\}$ and four computed answers in KD for $P \cup \{G'\}$: $\{x/husband(Jane)\}$, $\{x/husband(Mary)\}$, $\{x/Tom\}$, and $\{x/Jane\}$. Here is one of the SLD-refutations of $P \cup \{G'\}$ in KD :

Goals	Input Clauses / Operators	Mgu's
$\leftarrow \Diamond likes\_football(x)$		
$\leftarrow \langle X\rangle likes\_football(x)$	backward labeling	
$\leftarrow \langle X\rangle woman(x_2), \langle X\rangle likes\_football(x_2)$	ϕ_1	$\{x/x_2\}$
$\leftarrow \Box woman(x_2), \langle X\rangle likes\_football(x_2)$	$\Box$-lifting	
$\leftarrow woman(x_4), \langle X\rangle likes\_football(x_4)$	ϕ_4	$\{x_2/x_4\}$
$\leftarrow \langle X\rangle likes\_football(Jane)$	ϕ_8	$\{x_4/Jane\}$
$\leftarrow \Box likes\_football(Jane)$	$\Box$-lifting	
$\leftarrow likes\_football(Jane)$	ϕ_6	ε
$\diamond$	ϕ_{10}	ε

7 Soundness and Completeness of SLD-Resolution

7.1 Soundness

Every computed answer in L for $P \cup \{G\}$ is expected to be a correct answer in L for $P \cup \{G\}$. As we will see, this is true for G being an MProlog goal, however, in general we have a weaker result. The reason is that, despite that $\Box\alpha$ contains more information than $\Diamond\alpha$ (in serial modal logics), it does not contain more information than $\langle E\rangle\alpha$.

Lemma 7.1 *Let P be an MProlog program and $G = \leftarrow \alpha_1, \ldots, \alpha_k$ a goal. Then for every computed answer θ in L for $P \cup \{G\}$ there exists a goal $G' = \leftarrow \alpha'_1, \ldots, \alpha'_k$ such that α'_i is a $\Box$-lifting form of α_i, for $1 \leq i \leq k$, and $P \models_L \forall_c((\alpha'_1 \wedge \ldots \wedge \alpha'_k)\theta)$.*

Proof. Let the refutation of $P \cup \{G\}$ in L consist of a sequence $G_0 = G, G_1, \ldots, G_n$ of goals, a sequence $\phi_{i_1}, \ldots, \phi_{i_h}$ of variants of program clauses of P and a sequence $\theta_{i_1}, \ldots, \theta_{i_h}$ of mgu's. Let θ be the computed answer. We prove the result by induction on n.

Let M be an arbitrary L-model of P.

Suppose that $n = 1$. This means that $G = \leftarrow \alpha_1$, with $\alpha_1 = \triangle A'$ and $|\triangle| = s$, P has a clause $\phi_1 = \Box^t(A \leftarrow)$ such that $\Box^s(A \leftarrow)$ is its L-instance, and the empty clause $\diamond$ is an L-resolvent of G and ϕ_1, where A' is the atom being unified with the forward labeled form of A. We have $P \models_L \forall(\Box^s A)$. If A' is of the form $\Box E$ or E, then $A'\theta_1 = A\theta_1$, and $P \models_L \forall(\Box^s A'\theta_1)$. Suppose that $A' = \langle F\rangle E$ or $A' = \langle X\rangle E$. Thus A is of the form $\Diamond E'$. Let $A'' = \langle E'\rangle E'$ (the forward labeled form of A). We have $A'\theta_1 = A''\theta_1 = \langle E''\rangle E''$ for some E''. Since $P \models_L \forall(\Box^s A)$, we have $P \models_L \forall(\Box^s \Diamond E'')$. It follows that $P \models_L \forall_c(\Box^s \langle E''\rangle E'')$, hence $P \models_L \forall_c(\Box^s A'\theta_1)$.

Next suppose that the result holds for computed answers which come from refutations of length less than n. There are the following cases: G_1 is an L-resolvent of G and some program clause of P, G_1 is derived from G using $rSat_L$ or rNF_L, or G_1 is a backward labeled form or a $\Box$-lifting form of G. The last case (G_1 is a backward labeled form or a $\Box$-lifting form of G) is trivial.

Consider the first case. Suppose that G_1 is derived in L from G and a program clause $\phi = \Box^s(A \leftarrow B_1, \ldots, B_m)$ $(m \geq 0)$, the selected atom is $\alpha_i = \triangle A'$, and A' is the atom being unified with the forward labeled form of A. We have

$$G_1 = \leftarrow (\alpha_1, \ldots, \alpha_{i-1}, \triangle B_1, \ldots, \triangle B_m, \alpha_{i+1}, \ldots, \alpha_k)\theta_1$$

By inductive assumption, there exists a goal

$$G'_1 = \leftarrow (\alpha'_1, \ldots, \alpha'_{i-1}, \triangle_1 B'_1, \ldots, \triangle_m B'_m, \alpha'_{i+1}, \ldots, \alpha'_k)\theta_1$$

such that

$$P \models_L \forall_c((\alpha'_1 \wedge \ldots \wedge \alpha'_{i-1} \wedge \triangle_1 B'_1 \wedge \ldots \wedge \triangle_m B'_m \wedge \alpha'_{i+1} \wedge \ldots \wedge \alpha'_k)\theta)$$

where α'_j is a $\Box$-lifting form of α_j, for $1 \leq j \leq k$ and $j \neq i$, and $\triangle_j B'_j$ is a $\Box$-lifting form of $\triangle B_j$ with $|\triangle_j| = |\triangle|$, for $1 \leq j \leq m$. Let σ be a $\Diamond$-realization function such that $M, \sigma \models \phi$, where ϕ is the right formula in the above satisfaction. We have $M, \sigma \models \forall_c((\triangle_1 B'_1 \wedge \ldots \wedge \triangle_m B'_m)\theta)$ if $m > 0$. Let $\triangle'$ be the most general instance of $\triangle_1$, $\ldots$, $\triangle_m$ if $m > 0$, and be $\Box^t$ otherwise, where $t = |\triangle|$. Thus $\triangle'$ is a $\Box$-lifting form of $\triangle$, and $M, \sigma \models \forall_c((\triangle' B'_1 \wedge \ldots \wedge \triangle' B'_m)\theta)$ if $m > 0$.

Hence $M, \sigma \vDash \forall_c((\triangle' A)\theta)$. Let A'' be the forward labeled form of A. The partial function σ can be extended to σ' such that $M, \sigma' \vDash \forall_c((\triangle' A'')\theta)$. Since $A'\theta_1 = A''\theta_1$, we have

$$M, \sigma' \vDash \forall_c((\alpha'_1 \wedge \ldots \wedge \alpha'_{i-1} \wedge \triangle' A' \wedge \alpha'_{i+1} \wedge \ldots \wedge \alpha'_k)\theta).$$

Since M is an arbitrary L-model of P, we obtain

$$P \vDash_L \forall_c((\alpha'_1 \wedge \ldots \wedge \alpha'_{i-1} \wedge \triangle' A' \wedge \alpha'_{i+1} \wedge \ldots \wedge \alpha'_k)\theta).$$

Next suppose that G_1 is derived from G using $rSat_L$ or rNF_L – by replacing α_i by $\alpha = rSat_L(\alpha_i)$ or $\alpha = rNF_L(\alpha_i)$. By the inductive assumption, there exists a goal

$$G'_1 = \leftarrow \alpha'_1, \ldots, \alpha'_{i-1}, \alpha', \alpha'_{i+1}, \ldots, \alpha'_k$$

such that

$$P \vDash_L \forall_c((\alpha'_1 \wedge \ldots \wedge \alpha'_{i-1} \wedge \alpha' \wedge \alpha'_{i+1} \wedge \ldots \wedge \alpha'_k)\theta)$$

where α'_j is a $\Box$-lifting form of α_j, for $1 \leq j \leq k$ and $j \neq i$, and α' is a $\Box$-lifting form of α. There exists a $\Diamond$-realization function σ such that:

$$M, \sigma \vDash \forall_c((\alpha'_1 \wedge \ldots \wedge \alpha'_{i-1} \wedge \alpha' \wedge \alpha'_{i+1} \wedge \ldots \wedge \alpha'_k)\theta)$$

Consider the case when $\alpha = rSat_L(\alpha_i)$. It is sufficient to show that there exists an atom α'_i being a $\Box$-lifting form of α_i such that $M, \sigma \vDash \forall_c((\alpha'_i)\theta)$. A detailed proof should consider all possibilities of α. Here we give only some representatives:

- Case $L \in \{T, B, S4\}$, $\alpha_i = \triangle\beta$, and $\alpha = \triangle\Box\beta$: Let $\alpha' = \triangle'\Box\beta'$. Since M is a T-model and $M, \sigma \vDash \forall_c(\alpha'\theta)$, it follows that $M, \sigma \vDash \forall_c((\triangle'\beta')\theta)$. Choose $\alpha'_i = \triangle'\beta'$.
- Case $L \in \{KDB, B\}$, $\alpha_i = \triangle\beta$, and $\alpha = \triangle\langle X\rangle\Box\beta$ for a fresh atom variable X: Let $\alpha' = \triangle'\nabla\Box\beta'$. Since M is a KDB-model and $M, \sigma \vDash \forall_c(\alpha'\theta)$, it follows that $M, \sigma \vDash \forall_c((\triangle'\beta')\theta)$. Choose $\alpha'_i = \triangle'\beta'$.
- Case $L = KD5$, $\alpha_i = \Box\Box\Diamond E$, and $\alpha = \Box\langle X\rangle E$ for a fresh atom variable X: Since M is a $KD5$-model and $M, \sigma \vDash \forall_c(\alpha'\theta)$, it follows that $M, \sigma \vDash \forall_c(\Box\Box\Diamond E\theta)$. Choose $\alpha'_i = \Box\Box\Diamond E$.

Now consider the case when $L \in \{KD5, KD45, S5\}$ and $\alpha = rNF_L(\alpha_i)$. It is sufficient to show that there exist an atom α'_i being a $\Box$-lifting form of α_i and a $\Diamond$-realization function σ' being an extension of σ such that $M, \sigma' \vDash \forall_c(\alpha'_i\theta)$. There are the following cases to consider:

- Case $L \in \{KD45, S5\}$, $\alpha_i = \nabla E$ and $\alpha = \langle X\rangle\nabla E$ for a fresh atom variable X: If ∇ is of the form $\Box$ or $\Diamond$, then from $M, \sigma \vDash \forall_c(\alpha'\theta)$ we can derive $M, \sigma \vDash \forall_c(\nabla E\theta)$ and choose $\alpha'_i = \nabla E$, $\sigma' = \sigma$. Suppose that ∇ is of the form $\langle F\rangle$ (resp. $\langle Y\rangle$). Let $1 \leq j < n$ be the least index such that: i) the atom in G_j corresponding to α is the selected atom of G_j and is of the form $(\nabla'\nabla E)\delta$, where δ is the composition of the substitutions used in the derivation of $G_1, \ldots, G_j$; ii) G_{j+1} is not obtained from G_j by changing the selected atom to $(\Box\nabla E)\delta$. Let $\theta = \delta\gamma$. There are only the following subcases:
  - Case G_{j+1} is a $\Box$-lifting form of G_j and the selected atom of G_j is changed to $(\nabla''\Box E)\delta$: By the inductive assumption, we can derive $M, \sigma \vDash \forall_c((\Diamond\Box E\delta)\gamma)$, and hence $M, \sigma \vDash \forall_c(\Box E\theta)$. Choose $\alpha'_i = \Box E$ and $\sigma' = \sigma$.
  - Case G_{j+1} is an L-resolvent of G_j and some input clause: Because $(\nabla'\nabla E)\delta$ is used as a selected atom for deriving a resolvent in the SLD-refutation of $P \cup \{G_j\}$ in L with computed answer γ, we have $F\theta = E\theta$ (resp. $Y\theta = E\theta$). Let σ' be an extension of σ such that if $\sigma(\tau, \langle E'\rangle)$ is not defined and $M, \tau \vDash \Diamond E'$ then $\sigma'(\tau, \langle E'\rangle)$ is some world w of M with the property that $M, w \vDash E'$, where τ is the actual world of M and E' is an arbitrary ground classical atom. Let $\alpha' = \nabla_1\nabla_2 E$. With so defined σ' and the mentioned property of θ, we derive $M, \sigma' \vDash \forall_c((\nabla_2 E)\theta)$ from $M, \sigma \vDash \forall_c((\nabla_1\nabla_2 E)\theta)$. Choose $\alpha'_i = \nabla_2 E$.

- Case $L = KD5$, $\alpha_i = \Box\nabla E$ and $\alpha = \langle X\rangle\nabla E$ for a fresh atom variable X: If ∇ is of the form $\Box$ or $\Diamond$ then choose $\alpha_i' = \alpha_i$ and $\sigma' = \sigma$. Suppose that ∇ is of the form $\langle F\rangle$ (resp. $\langle Y\rangle$). Let j, δ, γ be defined in the same way as for the case $L \in \{KD45, S5\}$. Let the selected atom of G_j be the atom corresponding to α and be $(\nabla'\nabla E)\delta$. Similarly as for the case $L \in \{KD45, S5\}$, there are the following subcases:

  - Case G_{j+1} is a $\Box$-lifting form of G_j and the selected atom of G_j is changed to $(\nabla''\Box E)\delta$: By the inductive assumption, we can derive $M, \sigma \vDash \forall_c(\Diamond\Box E\theta)$, and hence $M, \sigma \vDash \forall_c(\Box\Box E\theta)$. Choose $\alpha_i' = \Box\Box E$ and $\sigma' = \sigma$.
  - Case G_{j+1} is an L-resolvent of G_j and some input clause: Because $(\nabla'\nabla E)\delta$ is used as a selected atom for deriving a resolvent in the SLD-refutation of $P \cup \{G_j\}$ in L with computed answer γ, either $\nabla' = \Box$ or $F\theta = E\theta$ (resp. $Y\theta = E\theta$).

    Suppose that $\nabla' = \Box$. By the inductive assumption, there exist σ'' and a $\Box$-lifting form α_i' of $\Box\nabla E$ (note that $\alpha_i'\delta$ is then a $\Box$-lifting form of the selected atom of G_j) such that $M, \sigma'' \vDash \forall_c(\alpha_i'\delta\gamma)$. By analyzing this proof, without loss of generality we can assume that σ is an extension of σ''. Thus $M, \sigma \vDash \forall_c(\alpha_i'\theta)$. Choose $\sigma' = \sigma$.

    Now suppose that $F\theta = E\theta$ (resp. $Y\theta = E\theta$). Let σ' be an extension of σ such that if $\sigma(w, \langle E'\rangle)$ is not defined and $M, w \vDash \Diamond E'$ then $\sigma'(w, \langle E'\rangle)$ is some world u of M such that $R(w, u)$ and $M, u \vDash E'$, where R is the accessibility relation of M and w, E' are arbitrary. Let $\alpha' = \nabla_1\nabla_2 E$. With so defined σ' and the mentioned property of θ, we derive $M, \sigma' \vDash \forall_c((\Box\nabla_2 E)\theta)$ from $M, \sigma \vDash \forall_c((\nabla_1\nabla_2 E)\theta)$. Choose $\alpha_i' = \Box\nabla_2 E$.
- Case $L = KD5$, $\alpha_i = \Box\nabla E$ and $\alpha = \Box\langle X\rangle\nabla E$ for a fresh atom variable X: The assertion holds by a similar argumentation as for the above case.

Thus we have completed the proof. □

As a corollary of the above lemma, soundness of SLD-resolution for MProlog is established.

Theorem 7.2 (Soundness of SLD-Resolution) *Let P be an MProlog program and G an MProlog goal. Then every computed answer in L for $P \cup \{G\}$ is a correct answer in L for $P \cup \{G\}$.*

Proof. Let $G = \leftarrow \alpha_1, \ldots, \alpha_k$, where each α_i is of the form $\Box^h E$ or $\Box^h\Diamond E$, with $h \geq 0$. Let θ be a computed answer in L for $P \cup \{G\}$. Since L is a serial modal logic, by Lemma 7.1, we have $P \vDash_L \forall_c((\alpha_1 \wedge \ldots \wedge \alpha_k)\theta)$. It follows that $P \vDash_L \forall((\alpha_1 \wedge \ldots \wedge \alpha_k)\theta)$ (here we assume there are enough constant symbols, for example, infinitely many). Hence θ is a correct answer in L for $P \cup \{G\}$. □

7.2 Completeness

We use a standard method to prove completeness of our SLD-resolution system (cf. [24, 23]). In general, completeness of a resolution calculus is first proved for the ground version and then lifted to the case with variables. The flow of this subsection follows Lloyd [24]. The proofs for Lemmas 7.5, 7.6, 7.11, and Theorem 7.12 are similar to the ones given for classical logic programming in Lloyd's book and are presented in the Appendix.

We first give two lemmas concerning properties of $rSat_L$ and rNF_L, which can be verified in a straightforward way.

Lemma 7.3 *Let $\beta = rSat_L(\alpha\theta)$ (resp. $\beta = rNF_L(\alpha\theta)$). Then there exists an atom β' such that $\beta' = rSat_L(\alpha)$ (resp. $\beta' = rNF_L(\alpha)$) and $\beta = \beta'\theta$.*

Lemma 7.4 *Let $\alpha \in Sat_L(\{\beta\})$ or $\alpha \in NF_L(\{\beta\})$. Then there exists an atom β' and a substitution θ such that $\beta = \beta'\theta$, the domain of θ consists of fresh atom variables, and β' can be derived from α using the backward labeling operator, the $\Box$-lifting operator, $rSat_L$ and rNF_L.*

We now define unrestricted SLD-refutations and give the *mgu lemma* and the *lifting lemma*, whose proofs are given in the Appendix.

Definition 7.1 An *unrestricted SLD-refutation* in L is an SLD-refutation in L, except that we drop the requirement that the substitutions θ_{i_j} be most general unifiers. They are only required to be unifiers.

Lemma 7.5 (Mgu Lemma) *Let P be an MProlog program and G a goal. Suppose that $P \cup \{G\}$ has an unrestricted SLD-refutation in L. Then $P \cup \{G\}$ has an SLD-refutation in L of the same length and the same sequence of indexes of mgu's such that, if $\theta_{i_1}, \ldots, \theta_{i_h}$ are the unifiers from the unrestricted refutation and $\theta'_{i_1}, \ldots, \theta'_{i_h}$ are mgu's from the refutation, then there exists a substitution γ such that $\theta_{i_1} \ldots \theta_{i_h} = \theta'_{i_1} \ldots \theta'_{i_h} \gamma$.*

Lemma 7.6 (Lifting Lemma) *Let P be an MProlog program, G a goal and θ a substitution. Suppose there exists an SLD-refutation of $P \cup \{G\theta\}$ in L such that the variables of the input clauses are distinct from the variables in G and θ. Then there exists an SLD-refutation of $P \cup \{G\}$ in L of the same length and the same sequence of indexes of mgu's such that, if $\theta_{i_1}, \ldots, \theta_{i_h}$ are the mgu's from the refutation of $P \cup \{G\theta\}$ and $\theta'_{i_1}, \ldots, \theta'_{i_h}$ are the mgu's from the refutation of $P \cup \{G\}$, then there exists a substitution γ such that $\theta\theta_{i_1} \ldots \theta_{i_h} = \theta'_{i_1} \ldots \theta'_{i_h} \gamma$.*

Lemma 7.7 (Weak Lifting Lemma) *Let P be an L-MProlog program, G a goal, and θ a substitution. Suppose there exists an SLD-refutation of $P \cup \{G\theta\}$ in L. Then there exists an SLD-refutation of $P \cup \{G\}$ in L.*

Proof. Change each variable x occurring in the input clauses of the refutation of $P \cup \{G\theta\}$ by x' that does not occur in G and θ. Then recompute the refutation. The resulting derivation will still be an SLD-refutation for $P \cup \{G\theta\}$ (possibly with different used mgu's and a different computed answer). Then apply the lifting lemma 7.6 for the new refutation. □

An essential part of the completeness proof is the following lemma, which can be viewed as a proof of completeness for the ground version.

Lemma 7.8 *Let P be an MProlog program and $\alpha \in I_{L,P}$. Then $P \cup \{\leftarrow \alpha\}$ has an SLD-refutation in L.*

Proof. We prove by induction on n that if $\alpha \in T_{L,P} \uparrow n$ then $P \cup \{\leftarrow \alpha\}$ has an SLD-refutation in L. This assertion obviously holds for $n = 0$ since $T_{L,P} \uparrow 0 = \emptyset$.

Suppose that the assertion holds for $(n-1)$ in the place of n. Let $\alpha \in T_{L,P} \uparrow n$. There exist a program clause $\phi = \Box^l(A \leftarrow B_1, \ldots, B_k)$ of P with $k \geq 0$, ground atoms $\gamma_1, \ldots, \gamma_k \in T_{L,P} \uparrow (n-1)$ and ground atoms $\beta_1, \ldots, \beta_k$, α' such that:

- $\beta_i \in Sat_L(\{\gamma_i\})$, for $1 \leq i \leq k$;
- there exists a substitution θ such that $B_i\theta$ is an instance of B'_i and $\beta_i = \triangle_i B'_i$, for $1 \leq i \leq k$;
- $\triangle$ is the most general instance of $\triangle_1, \ldots, \triangle_k$, $|\triangle| = s$ and $\Box^s(A \leftarrow B_1, \ldots, B_k)$ is an L-instance of ϕ;
- $\alpha' = \triangle A'\theta$, where A' is the forward labeled form of A;
- $\{\alpha\} = NF_L(\{\alpha'\})$ if $L \in \{KD5, KD45, S5\}$, and $\alpha = \alpha'$ otherwise.

By Lemma 7.4, there exist atoms α'', $\gamma'_1, \gamma'_2, \ldots, \gamma'_k$, and ground substitutions $\delta_0, \ldots, \delta_k$ with disjoint domains such that:

- α'' can be derived from α using the backward labeling operator, the $\Box$-lifting operator, $rSat_L$ and rNF_L, and furthermore, $\alpha' = \alpha''\delta_0$,
- γ'_i can be derived from β_i using the backward labeling operator, the $\Box$-lifting operator, $rSat_L$ and rNF_L, and furthermore, $\gamma_i = \gamma'_i\delta_i$, for $1 \leq i \leq k$.

Let $\delta = \delta_1 \dots \delta_k$ if $k > 0$, and $\delta = \varepsilon$ otherwise. By the inductive assumption, $P \cup \{\leftarrow \gamma_i\}$ has a refutation in L, for $1 \leq i \leq k$. Since $\gamma_i'\delta = \gamma_i'\delta_i = \gamma_i$, it follows that $P \cup \{\leftarrow \gamma_i'\delta\}$ has a refutation in L. Hence $P \cup \{\leftarrow (\gamma_1', \dots, \gamma_k')\delta\}$ has a refutation in L, since $\gamma_i'\delta$ are ground. By the weak lifting lemma, $P \cup \{\leftarrow \gamma_1', \dots, \gamma_k'\}$ has a refutation in L. Because γ_i' can be derived from β_i using the backward labeling operator, the $\Box$-lifting operator, $rSat_L$ and rNF_L, it follows that $P \cup \{\leftarrow \beta_1, \dots, \beta_k\}$ has a refutation in L.

For $1 \leq i \leq k$, if B_i' is of the form $\langle F\rangle E$ then let $\beta_i' = \triangle_i\langle X\rangle E$ and $\theta_i = \{X/F\}$, where X is a fresh atom variable; else let $\beta_i' = \beta_i$ and $\theta_i = \varepsilon$. Let $\theta' = \theta_1 \dots \theta_k$ if $k > 0$, and $\theta' = \varepsilon$ otherwise. Since $\beta_i = \beta_i'\theta'$, $P \cup \{\leftarrow (\beta_1', \dots, \beta_k')\theta'\}$ has a refutation in L. Hence, by the weak lifting lemma, $P \cup \{\leftarrow \beta_1', \dots, \beta_k'\}$ has a refutation in L. Therefore, by applying the backward labeling operator and the $\Box$-lifting operator, $P \cup \{\leftarrow \triangle_1 B_1\theta, \dots, \triangle_k B_k\theta\}$ has a refutation in L. Hence, by applying the $\Box$-lifting operator, $P \cup \{\leftarrow \triangle B_1\theta, \dots, \triangle B_k\theta\}$ has a refutation in L. The goal $\leftarrow \triangle B_1\theta, \dots, \triangle B_k\theta$ is an unrestricted L-resolvent of $\leftarrow \alpha'$ and ϕ. Hence, by the mgu lemma, $P \cup \{\leftarrow \alpha'\}$ has a refutation in L. This means that $P \cup \{\leftarrow \alpha''\delta_0\}$ has a refutation in L. By the weak lifting lemma, $P \cup \{\leftarrow \alpha''\}$ has a refutation in L. Because α'' can be derived from α using the backward labeling operator, the $\Box$-lifting operator, $rSat_L$ and rNF_L, we conclude that $P \cup \{\leftarrow \alpha\}$ has a refutation in L. $\Box$

Corollary 7.9 *Let P be an MProlog program and $\alpha \in Sat_L(I_{L,P})$. Then $P \cup \{\leftarrow \alpha\}$ has an SLD-refutation in L.*

Proof. There exists $\beta \in I_{L,P}$ such that $\alpha \in Sat_L(\{\beta\})$. Consequently, there exist an atom β' and a substitution θ such that $\beta = \beta'\theta$ and β' can be derived from α using the backward labeling operator, the $\Box$-lifting operator, $rSat_L$ and rNF_L. Since $\beta \in I_{L,P}$, by Lemma 7.8, $P \cup \{\leftarrow \beta\}$ has a refutation in L. This means that $P \cup \{\leftarrow \beta'\theta\}$ has a refutation in L. By the weak lifting lemma, $P \cup \{\leftarrow \beta'\}$ has a refutation in L. Consequently, $P \cup \{\leftarrow \alpha\}$ has a refutation in L. $\Box$

If α is a classical atom such that $M_{L,P} \vDash \alpha$, then $\alpha \in Ext_L(I_{L,P}) \subseteq Sat_L(I_{L,P})$, and hence $P \cup \{\leftarrow \alpha\}$ has an SLD-refutation in L. This also holds for the case when α is a ground MProlog goal atom.

Lemma 7.10 *Let P be an MProlog program and α a ground MProlog goal atom such that $M_{L,P} \vDash \alpha$. Then $P \cup \{\leftarrow \alpha\}$ has an SLD-refutation in L.*

Proof. In some cases, by Corollary 7.9, it is sufficient to show that $\alpha \in Sat_L(I_{L,P})$. Consider the case when α is of the form $\Box^s E$, $s \geq 0$. Since $M_{L,P} \vDash \alpha$, it follows that $\alpha \in Ext_L(I_{L,P})$, and hence $\alpha \in Sat_L(I_{L,P})$.

Now suppose that $\alpha = \Box^s\Diamond E$, $s \geq 0$. Note that $s = 0$ for $L \in \{KD45, S5\}$ and $s \leq 1$ for $L = KD5$. Let $\langle W, \tau, R, H\rangle$ be the standard L-model graph of $I_{L,P}$. Let w be the world $\langle\top\rangle \dots \langle\top\rangle$ with length s. Since $M_{L,P} \vDash \alpha$, we have $M_{L,P}, w \vDash \Diamond E$, and hence there are (only) the following cases:

- Case $E \in H(u)$ for some $u = w\langle F\rangle$: By Lemma 4.1, $\langle\top\rangle^s\langle F\rangle E$ is an instance of some atom from $Ext_L(I_{L,P})$. It follows that either $\Box^{s+1}E$ or $\Box^s\langle F\rangle E$ (with $F \neq \top$) belongs to $Ext_L(I_{L,P})$. Denote the atom by α'. By Corollary 7.9, $P \cup \{\leftarrow \alpha'\}$ has a refutation in L. If $\alpha' = \Box^{s+1}E$, then by applying the $\Box$-lifting operator, $P \cup \{\leftarrow \alpha\}$ has a refutation in L. If $\alpha' = \Box^s\langle F\rangle E$, then by the weak lifting lemma, $P \cup \{\leftarrow \Box^s\langle X\rangle E\}$ with X being a fresh atom variable has a refutation in L. Hence, by applying the backward labeling operator, $P \cup \{\leftarrow \alpha\}$ has a refutation in L.

- Case $L \in \{T, B, S4, S5\}$ and $E \in H(w)$: By Lemma 4.1, $\langle\top\rangle^s E$ is an instance of some atom from $Ext_L(I_{L,P})$. It follows that $\Box^s E \in Ext_L(I_{L,P})$, and hence $\alpha \in Sat_L(I_{L,P})$.

- Case $L \in \{KDB, B\}$ and $E \in H(u)$, where $w = u\langle\top\rangle$: By Lemma 4.1, $\langle\top\rangle^{s-1}E$ is an instance of some atom from $Ext_L(I_{L,P})$. It follows that $\Box^{s-1}E \in Ext_L(I_{L,P})$, and hence $\alpha \in Sat_L(I_{L,P})$.

- Case $L \in \{KD4, S4\}$ and $E \in H(u)$ for some $u = w\langle F_1\rangle \dots \langle F_k\rangle$: By Lemma 4.1, $\langle\top\rangle^s\langle F_1\rangle \dots \langle F_k\rangle E$ is an instance of some atom from $Ext_L(I_{L,P})$. It follows that $\Box^s\langle F_1\rangle \dots \langle F_k\rangle E$ is an instance of some atom from $Ext_L(I_{L,P})$, and hence $\alpha \in Sat_L(I_{L,P})$.

- Case $L = KD5$, $s = 1$, and $E \in H(u)$ for some $u = \langle F \rangle$: By Lemma 4.1, $\langle F \rangle E$ is an instance of some atom from $Ext_L(I_{L,P})$, and hence $\alpha \in Sat_L(I_{L,P})$.

□

For the main theorem, we need another auxiliary lemma, whose proof uses Lemma 7.10 and is presented in the Appendix.

Lemma 7.11 *Let P be an MProlog program and α an MProlog goal atom. Suppose that $\forall(\alpha)$ is a logical consequence in L of P. Then there exists an SLD-refutation of $P \cup \{\leftarrow \alpha\}$ in L with the identity substitution as the computed answer.*

The following completeness theorem follows from the above lemma and the lifting lemma. Its proof is given in the Appendix.

Theorem 7.12 (Completeness of SLD-Resolution) *Let P be an MProlog program and G an MProlog goal. For every correct answer θ in L for $P \cup \{G\}$, there exists a computed answer γ in L for $P \cup \{G\}$ and a substitution δ such that $G\theta = G\gamma\delta$.*

8 Semantics for Programs in Almost Serial Modal Logics

A Kripke frame is said to be *flat* if there are no worlds reachable from the actual world. A frame is *serial* if for every one of its worlds there exists a world reachable from that world. A frame is *connected* if every one of its worlds is reachable directly or indirectly from the actual world or is the actual world itself. A Kripke model is said to be flat (resp. serial) if its frame is flat (resp. serial).

A modal logic L is called *almost serial* if every connected L-frame is either serial or flat. Given an almost serial modal logic L, there is a serial modal logic corresponding to it, denoted by LD, with the property that every serial L-frame is an LD-frame. The logics KB, $K5$, $K45$, and $KB5$ are almost serial, and their corresponding serial modal logics are respectively KDB, $KD5$, $KD45$, and $S5$. In this section, we investigate an SLD-resolution calculus for MProlog in the mentioned almost serial modal logics.

In [29], we showed that every positive propositional modal logic program P can be *characterized* in $L \in \{KB, K5, K45, KB5\}$ by one or two models (a least LD-model of P and a flat model) in the sense that a positive formula ϕ is a consequence of P in L iff ϕ is true in those models. This assertion still holds for the first-order case.

From now to the end of this section, by L we denote one of the logics KB, $K5$, $K45$, and $KB5$.

Definition 8.1 Let P be an MProlog program. By $Flat(P)$ we denote the classical positive program obtainable from P by: removing all clauses starting with $\Box$ (i.e. the ones with non-empty modal context), removing all clauses of the form $\Box E \leftarrow B_1, \ldots B_n$ $(n \geq 0)$, deleting all atoms of the form $\Box E$ and replacing all atoms of the form $\Diamond E$ by a special atom $\mathfrak{f}$ (which informally stands for falsity) in the remaining clauses. By $M_{Flat(P)}$ we denote the least (classical) Herbrand model of $Flat(P)$.

It is easily seen that if $\mathfrak{f} \in M_{Flat(P)}$, then P has no flat models and $M_{LD,P}$ is a least L-model of P. If $\mathfrak{f} \notin M_{Flat(P)}$, then P is characterized in L by $M_{LD,P}$ and $M_{Flat(P)}$, which means that for every positive ground formula ϕ, $P \models_L \phi$ iff ϕ is true in both $M_{LD,P}$ and $M_{Flat(P)}$.

Definition 8.2 Let $G = \leftarrow \alpha_1, \ldots, \alpha_k$ be an MProlog goal. If some of α_i are of the form $\Diamond E$, then by $Flat(G)$ we denote the goal $\leftarrow \mathfrak{f}$, else $Flat(G)$ is the classical goal obtainable from G by deleting all atoms starting with $\Box$.

If P is a classical logic program and G is a classical goal, then SLD-refutation of $P \cup \{G\}$ in the classical logic is defined in the usual way (see [24]). It can be also defined, for example, as an SLD-refutation of $P \cup \{G\}$ in KD with no occurrences of modal operators.

We now define SLD-refutation in $L \in \{KB, K5, K45, KB5\}$.

Definition 8.3 Let P be an MProlog program and G an MProlog goal. If $L \in \{K5, K45, KB5\}$, then let P' be the MProlog program in LD equivalent in LD to P, and G' the MProlog goal in LD equivalent in LD to G (see Definition 3.5). If L is KB then let $P' = P$ and $G' = G$. An *SLD-refutation* of $P \cup \{G\}$ in L is either:

- a composition of an SLD-refutation of $P' \cup \{G'\}$ in LD and an SLD-refutation of $Flat(P) \cup \{\leftarrow \mathfrak{f}\}$ in the classical logic, or
- a composition of an SLD-refutation of $P' \cup \{G'\}$ in LD and an SLD-refutation of $Flat(P) \cup \{Flat(G)\theta\}$ in the classical logic, where θ is the computed answer coming from the refutation of $P' \cup \{G'\}$ in LD.

By a composition of two refutations we mean the ordered pair of the refutations; the concatenation of the sequences of input clauses (resp. mgu's) from the refutations is treated as the sequence of input clauses (resp. mgu's) for the composition. Standardizing variables apart is required throughout the composition of refutations. The definition of computed answers is as usual. In the following we give a soundness theorem and a completeness theorem for SLD-resolution in L.

Theorem 8.1 *Let $L \in \{KB, K5, K45, KB5\}$, P be an MProlog program and G an MProlog goal. Then every computed answer in L for $P \cup \{G\}$ is a correct answer in L for $P \cup \{G\}$.*

The proof of this theorem is straightforward.

Theorem 8.2 *Let $L \in \{KB, K5, K45, KB5\}$, P be an MProlog program and G an MProlog goal. Let θ be a correct answer in L for $P \cup \{G\}$. Then there exist a computed answer γ in L for $P \cup \{G\}$ and a substitution δ such that $G\theta = G\gamma\delta$.*

Proof. If $L \in \{K5, K45, KB5\}$, then let P' be the MProlog program in LD equivalent in LD to P, and G' the MProlog goal in LD equivalent in LD to G. If L is KB then let $P' = P$ and $G' = G$.

Consider the case $\mathfrak{f} \in M_{Flat(P)}$. Thus $Flat(P) \cup \{\leftarrow \mathfrak{f}\}$ has an SLD-refutation in the classical logic. Besides, θ is a correct answer in LD for $P' \cup \{G'\}$. Since SLD-resolution in LD is complete, it follows that $P' \cup \{G'\}$ has an SLD-refutation in LD with computed answer γ and there exists a substitution δ such that $G'\theta = G'\gamma\delta$, which implies $G\theta = G\gamma\delta$. An appropriate composition of the two above mentioned refutations is an SLD-refutation of $P \cup \{G\}$ in L with computed answer γ.

Now suppose that $\mathfrak{f} \notin M_{Flat(P)}$. Thus θ is a correct answer in LD for $P' \cup \{G'\}$ and a correct answer in the classical logic for $Flat(P) \cup \{Flat(G)\}$. There exist a computed answer γ_1 in LD for $P' \cup \{G'\}$ and a substitution θ' such that $G'\theta = G'\gamma_1\theta'$, which implies $G\theta = G\gamma_1\theta'$. Since θ is a correct answer in the classical logic for $Flat(P) \cup \{Flat(G)\}$, $Flat(P) \cup \{Flat(G)\theta\}$ has an SLD-refutation in the classical logic with the identity substitution as the computed answer (see the proof of Theorem 8.6 of [24]). We have $Flat(G)\theta = Flat(G)\gamma_1\theta'$. Change each variable x occurring in the input clauses of the refutation of $Flat(P) \cup \{Flat(G)\theta\}$ by x' that does not occur in G, γ_1 and θ'. Then recompute the refutation in the way such that when unifying $\{x, x'\}$ the substitution $\{x'/x\}$ is used instead of $\{x/x'\}$. The resulting derivation will still be an SLD-refutation for $P \cup \{Flat(G)\gamma_1\theta'\}$ with the identity substitution as the computed answer. By the lifting lemma, $Flat(P) \cup \{Flat(G)\gamma_1\}$ has an SLD-refutation in the classical logic with computed answer γ_2, and there exists a substitution δ such that $\theta'\varepsilon = \gamma_2\delta$. Hence there exists an SLD-refutation of $P \cup \{G\}$ in L with computed answer $\gamma = \gamma_1\gamma_2$. Note that $G\theta = G\gamma\delta$. □

9 Related Work

In this section, we briefly discuss some related works. We refer the reader to Minker's work [25] for a good survey and the works by Lloyd [24] and Apt [4] for foundations of classical logic programming. As surveys on modal and temporal logic programming, there are the works by Orgun and Ma [32], and Fisher and Owens [15].

We have adopted from Lloyd's book [24] the clausal notation for programs and goals. Our presentation of SLD-resolution and the method for proving completeness are influenced by Lloyd's book. Leitsch' book [23] also helps us in understanding the nature of resolution calculi.

This work can be viewed as a continuation of our previous works [29, 30]. In [30], we introduced the modal query language MDatalog. An MDatalog program is an MProlog program without function symbols and with some constraints on variables. In the mentioned work, we gave algorithms to construct least models for MDatalog programs. Given an MDatalog program P, a least L-model for P can be effectively constructed, where L is any serial modal logic considered in this work. The key in building a least L-model for P is the task of connecting each newly created world to an empty world at the time of its creation. The fixpoint semantics for MProlog programs differs from the algorithm of constructing least models for MDatalog programs on the structure of objects: in the latter case it is model graphs, while in the former case it is model generators, which are set representations of model graphs. Our work [29] on constructing least models for propositional modal logic programs are influenced by the works on the complexity of (Horn) modal logics [11, 12, 19]. The Mints translation, which we studied from [22, 26], is very helpful for simplifying the form of modal logic programs.

In [10], Fariñas del Cerro proposed a modal logic programming system called Molog. The Molog language is similar to and as expressive as our eMProlog language. With Molog, the user can fix a modal logic and define or choose the rules to deal with modal operators. Molog can be viewed as a framework which can be instantiated with particular modal logics. Balbiani et al [5] gave a declarative semantics and an SLD-resolution for a class of Molog programs in the logics Q (KD in our notation), T and $S4$. To modal programs the authors associate a declarative semantics represented by a tree which is defined as the limit of a certain transformation on modal programs. The fixpoint represents a minimal Kripke model of the program. There is a common point between that work and our work; in both of the works, labeled modal operators are used to convert $\langle t\rangle(\phi\wedge\psi)$ to $\langle t\rangle\phi\wedge\langle t\rangle\psi$. Labeled modal operators in [5] come from Skolemization, and terms are used to label the $\Diamond$ operator. In our work, the labeling technique results from the technique of building model graphs, and we feel convenient to use classical atoms and atom variables to label the $\Diamond$ operator. Modal programs considered in [5] require a restriction that the $\Box$ operator does not occur in the bodies of modal clauses. This restriction is also required for goals. In the definition of the minimal Kripke model of a program [5], the technique of connecting each newly created world to an empty world at the time of its creation (or a similar one) is not used, hence although the minimal Kripke model of a program defined in [5] is minimal with respect to the restricted class of goals, in general it is not a least Kripke model of the program in the considered logic. We think that if the technique of using empty worlds is adopted, then the restriction on goals and bodies of modal clauses in [5] can be dropped and the semantics can be extended for the full language. The SLD-resolution calculus defined in [5] is not as declarative as ours, because a direct resolvent of a goal G and a program clause ϕ is not specified by the syntax of G and ϕ but it needs inference steps.

As an extension of Molog, the *Toulouse Inference Machine* (TIM) [6] (together with an abstract machine model called TARSKI for implementation [7]) makes it possible for a user to select clauses which cannot exactly unify with the current goal, but just resemble it in some way. TIM is also a general framework for modal logic programming.

Another approach to modal logic programming is based on the translation technique as in Ohlbach's resolution calculus for modal logics [31]. In Akama's work [2], modal programs are translated into programs of a two-sorted first-order logic by introducing world paths as extra parameters to function and predicate symbols[9]. A meta-interpreter is given for the execution of translated programs. The translation method does not require the axiomatization of the accessibility relations associated with modal operators such as $\Box$ and $\Diamond$ in modal logic programs, rather it directly encodes the accessibility relations into the unification algorithm. Considered modal logics are KD, T, KDB, B, $S4$, and $S5$. Although it is shown that a modal Herbrand property holds for translated modal formulae, the declarative semantics of modal logic programs are not given.

Debart et al [13] also used a translation technique for multimodal logic programming. In that work, the authors gave an automated theorem proving method called Σ–E-resolution for multimodal logics. The logics have a finite number of modal operators $\Box_i$ and $\Diamond_i$ of any type among KD, KT, KD4, KT4, KF, and interaction axioms of the form $\Box_i\phi\rightarrow\Box_j\phi$. Multimodal formulae are first translated into formulae of order-sorted equational theories, preserving satisfiability, and then Σ–E-resolution is used to show satisfiability of the translated formulae. Like the work by Akama [2],

[9]This paragraph and the preceding one are based on the survey by Orgun and Ma [32].

extra arguments (world paths) are added to function and predicate symbols. Translated formulae are called *path formulae*. A special unification algorithm is developed to deal with path formulae and the properties of modal operators. When path formulae are restricted to Horn clauses, a logic programming language called Pathlog is obtained. Debart et al [13] showed that Σ–E-resolution with the restriction to Horn clauses is a sound and complete procedure for Pathlog.

The method used by Akama [2] and Debart et al [13] speeds up the procedures of searching answers for queries. There are, however, some problems involved with it. Firstly, it seems hard to develop the least Kripke model semantics for Pathlog programs (even in monomodal logics). Secondly, path formulae are less intuitive for users than modal formulae in interactive and debugging modes of programming. Finally, because unification algorithms for path formulae either are nondeterministic or may give many unifiers, adapting the depth-first search strategy is more complicated.

10 Conclusions

In summary, we have proposed the modal logic programming language MProlog, which is as expressive as the general modal Horn fragment, and have given a fixpoint semantics and an SLD-resolution calculus for MProlog programs in the serial modal logics KD, T, KDB, B, $K4$, $S4$, $KD5$, $KD45$, and $S5$. There is a close relationship between the semantics and the calculus. Our SLD-resolution calculus has been designed with a similar style as for classical logic programming and shown to be sound and complete. The calculus has been extended also for MProlog programs in the almost serial modal logic KB, $K5$, $K45$, and $KB5$.

In our technical report [28], we have justified that independence of the computation rule also holds for our SLD-resolution calculi. In that report we have also presented another approach to dealing with programming in the logics $KD5$, $KD45$ and $S5$, which is based on translating programs in those logics into a simpler form that allows more efficient SLD-resolution calculi.

Our work brings new results and new methods for modal logic programming. It extends the most important results of classical logic programming for modal logic programming in a modal and systematic way. Our results have been established for a large class of monomodal logics: from the class of logics that are obtainable from K by adding an arbitrary combination of the axioms D, T, B, 4 and 5 only the logics K and $K4$ are left[10]. Our results and methods can be used as a basis for further works in modal logic programming, e.g. to develop semantics and SLD-resolution calculi for multimodal logic programming, computing methods for (multi)modal deductive databases, or SLDNF-resolution calculi for (multi)modal logic programs with negation.

Acknowledgements

The author would like to express his thanks to professors Andrzej Szałas and Witold Łukaszewicz for reading the draft of this paper and giving helpful comments.

References

[1] Abadi, M., Manna, Z.: Temporal Logic Programming, *Journal of Symbolic Computation*, **8**, 1989, 277–295.

[2] Akama, S.: A Meta-Logical Foundation of Modal Logic Programming. 1-20-1, Higashi-Yurigaoka, Asao-ku, Kawasaki-shi, 215, Japan, December 1989.

[3] Aoyagi, T., Fujita, M., Moto-oka, T.: Temporal Logic Programming Language Tokio, *E. Wada, editor, Logic Programming'85, LNCS 221*, Springer-Verlag, 1986.

[4] Apt, K.: Logic Programming, in: *Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics* (J. van Leeuwen, Ed.), Elsevier, 1990.

[10] Positive propositional logic programs in $L \in \{K, K4\}$ cannot be characterized by a finitely bounded number of L-models (see [29]).

[5] Balbiani, P., Fariñas del Cerro, L., Herzig, A.: Declarative Semantics for Modal Logic Programs, *Proceedings of the 1988 International Conference on Fifth Generation Computer Systems*, ICOT, 1988.

[6] Balbiani, P., Herzig, A., Lima-Marques, M.: TIM: The Toulouse Inference Machine for Non-Classical Logic Programming, *PDK'91: International Workshop on Processing Declarative Knowledge*, Springer-Verlag, 1991.

[7] Balbiani, P., Herzig, A., Lima-Marques, M.: Implementing Prolog Extensions: A Parallel Inference Machine, *Proceedings of the 1992 International Conference on Fifth Generation Computer Systems*, ICOT, 1992.

[8] Barringer, H., Fisher, M., Gabbay, D., Gough, G., Owens, R.: METATEM: A Framework for Programming in Temporal Logic, *Proceedings of REX Workshop on Stepwise Refinement of Distributed Systems: Models, Formalisms, Correctness, LNCS 430*, Springer-Verlag, 1989.

[9] Brzoska, C.: Temporal Logic Programming with Metric and Past Operators, *Executable Modal and Temporal Logics, IJCAI'93 workshop, M. Fisher and R. Owens (eds.)*, Springer, 1995.

[10] Fariñas del Cerro, L.: MOLOG: A System that Extends PROLOG with Modal Logic, *New Generation Computing*, **4**, 1986, 35–50.

[11] Fariñas del Cerro, L., Penttonen, M.: A Note on the Complexity of the Satisfiability of Modal Horn Clauses, *Logic Programming*, **4**, 1987, 1–10.

[12] Chen, C., Lin, I.: The Computational Complexity of the Satisfiability of Modal Horn Clauses for Modal Propositional Logics, *Theoretical Computer Science*, **129**, 1994, 95–121.

[13] Debart, F., Enjalbert, P., Lescot, M.: Multimodal Logic Programming Using Equational and Order-Sorted Logic, *Theoretical Computer Science*, **105**, 1992, 141–166.

[14] Fisher, M., Owens, R.: From the Past to the Future: Executing Temporal Logic Programs, *Proceedings of Logic Programming and Automated Reasoning, LNCS 624*, Springer-Verlag, 1992.

[15] Fisher, M., Owens, R.: An Introduction to Executable Modal and Temporal Logics, *Executable Modal and Temporal Logics, IJCAI'93 workshop, M. Fisher and R. Owens (eds.)*, Springer, 1995.

[16] Gabbay, D.: Modal and Temporal Logic II (A Temporal Logic Machine), in: *Logic Programming–Expanding the Horizon* (T. Dodd, R. Owens, S. Torrance, Eds.), Intellect Books Ltd, 1991.

[17] Garson, J.: Quantification in Modal Logic, in: *Handbook of Philosophical Logic, Volume II* (F. Guenthner, D. Gabbay, Eds.), 1999, 249–307.

[18] Hale, R.: Temporal Logic Programming, in: *Temporal Logics and Their Applications* (A. Galton, Ed.), Academic Press, 1987, 91–119.

[19] Halpern, J.: The Effect of Bounding the Number of Primitive Propositions and the Depth of Nesting on the Complexity of Modal Logic, *Artificial Intelligence*, **75**, 1995.

[20] Hrycej, T.: Temporal Prolog, *Yves Kodratoff, editor, Proceedings of the European Conference on Artificial Intelligence*, Pitman Publishing, 1988.

[21] Hrycej, T.: A Temporal Extension of Prolog, *The Journal of Logic Programming*, **15**(1&2), 1993, 113–145.

[22] Hudelmaier, J.: Improved Decision Procedures for the Modal Logics K, T, S4, *H. Kleine Büning, editor, Proceedings of CSL'95, LNCS 1092*, Springer, 1996.

[23] Leitsch, A.: *The Resolution Calculus*, Springer, 1997.

[24] Lloyd, J.: *Foundations of Logic Programming, Second Edition*, Springer-Verlag, 1987.

[25] Minker, J.: Logic and Databases: A 20-Year Retrospective, *International Workshop LID'96*, Verlag Springer, 1996.

[26] Mints, G.: Gentzen-type Systems and Resolution Rules, *P.Martin-Löf, G. Mints (eds.): COLOG-88, LNCS 417*, Springer, 1988.

[27] Moszkowski, B.: *Executing Temporal Logic Programs*, Cambridge University Press, 1986.

[28] Nguyen, L.: *A Fixpoint Semantics and an SLD-Resolution Calculus for Modal Logic Programs*, Technical Report TR 01-02 (265), Institute of Informatics, Warsaw University, 2001 (last revised January, 2003). Available on http://www.mimuw.edu.pl/~nguyen/papers.html.

[29] Nguyen, L.: Constructing the Least Models for Positive Modal Logic Programs, *Fundamenta Informaticae*, **42**(1), 2000, 29–60.

[30] Nguyen, L.: The Modal Query Language MDatalog, *Fundamenta Informaticae*, **46**(2001), 2001, 315–342.

[31] Ohlbach, H.: A Resolution Calculus for Modal Logics, *Proceedings of CADE-88, LNCS310*, Springer, 1988.

[32] Orgun, M., Ma, W.: An Overview of Temporal and Modal Logic Programming, *D.M. Gabbay and H.J. Ohlbach, editors, Proc. First Int. Conf. on Temporal Logic - LNAI 827*, Springer-Verlag, 1994.

[33] Orgun, M., Wadge, W.: *Chronolog: A Temporal Logic Programming Language and Its Formal Semantics*, Department of Computer Science, University of Victoria, Victoria, B.C., Canada, 1988.

[34] Rolston, D.: *Chronolog: A Pure Tense-Logic-Based Infinite-Object Programming Language*, Department of Computer Science and Engineering, Arizona State University, Tempe, Arizona, 1986.

Appendix

Proof of the mgu lemma 7.5

Let the unrestricted refutation of $P \cup \{G\}$ consist of a sequence $G_0 = G, G_1, \ldots, G_n$ of goals, a sequence $\phi_{i_1}, \ldots, \phi_{i_h}$ of variants of program clauses of P and a sequence $\theta_{i_1}, \ldots, \theta_{i_h}$ of unifiers. Let θ be the computed answer from the unrestricted refutation. We prove the result by induction on n.

Suppose that $n = 1$. This means that $G = \leftarrow \triangle A'$, with $|\triangle| = s$, the program has a clause $\phi_1 = \Box^t(A \leftarrow)$ such that $\Box^s(A \leftarrow)$ is its L-instance, and the empty clause $\diamond$ is an L-resolvent of G and ϕ_1 with unifier θ_1, where A' is the atom being unified with the forward labeled form of A. Suppose that θ'_1 is an mgu of A' and the forward labeled form of A. Then $\theta_1 = \theta'_1\gamma$ for some γ. Furthermore, $P \cup \{G\}$ has a refutation in L consisting of $G_0 = G$, $G_1 = \diamond$ with input clause ϕ_1 and mgu θ'_1.

Now suppose that the result holds for $n-1$. The cases when G_1 is a backward labeled form or a $\Box$-lifting form of G or is derived from G using $rSat_L$ or rNF_L are trivial. Let $G = \leftarrow \alpha_1, \ldots, \alpha_k$. Suppose that G_1 is derived from G and the input clause $\phi_1 = \Box^s(A \leftarrow B_1, \ldots, B_m)$ in L, the selected atom is $\alpha_i = \triangle A'$, and A' is the atom being unified with the forward labeled form of A. There exists an mgu θ'_1 for A' and the forward labeled form of A. We have $\theta_1 = \theta'_1\delta$ for some δ. Let G'_1 be the goal derived in the way as G_1 but with mgu θ'_1 instead of θ_1. We have $G_1 = G'_1\delta$. By Lemma 7.3, the derivation with the subsequence of goals $G_1, \ldots, G_{i_2-1}$ can be simulated by an SLD-derivation in L with a sequence of goals $G'_1, \ldots, G'_{i_2-1}$ such that $G_j = G'_j\delta$, for $1 \leq j \leq i_2 - 1$. Then G_{i_2} can be derived from G'_{i_2-1} and ϕ_{i_2} in L using the unifier $\delta\theta_{i_2}$. Thus $P \cup \{G\}$ has an unrestricted refutation in L consisting of $G_0 = G, G'_1, \ldots, G'_{i_2-1}, G_{i_2}, \ldots, G_n$ with input clauses $\phi_1, \phi_{i_2}, \ldots, \phi_{i_h}$ and unifiers $\theta'_1, \delta\theta_{i_2}, \theta_{i_3}, \ldots, \theta_{i_h}$. By the inductive assumption, $P \cup \{G'_1\}$ has a

refutation in L with mgu's $\theta'_{i_2}, \ldots, \theta'_{i_h}$ such that $\delta\theta_{i_2} \ldots \theta_{i_h} = \theta'_{i_2} \ldots \theta'_{i_h}\gamma$, for some γ. Thus $P \cup \{G\}$ has a refutation in L consisting of $G_0 = G$, $G'_1, \ldots, G'_n = \diamond$ with mgu's $\theta'_1, \theta'_{i_2} \ldots \theta'_{i_h}$ such that $\theta_1\theta_{i_2} \ldots \theta_{i_h} = \theta'_1\delta\theta_{i_2} \ldots \theta_{i_h} = \theta'_1\theta'_{i_2} \ldots \theta'_{i_h}\gamma$.

Proof of the lifting lemma 7.6
Let the refutation of $P \cup \{G\theta\}$ consist of a sequence $G_0 = G\theta, G_1, \ldots, G_n$ of goals, a sequence $\phi_{i_1}, \ldots, \phi_{i_h}$ of variants of program clauses of P and a sequence $\theta_{i_1}, \ldots, \theta_{i_h}$ of mgu's.

We can prove the lemma by induction on n, and with that way the case when G_1 is a backward labeled form or a $\Box$-lifting form of G is trivial, the case when G_1 is derived from G using $rSat_L$ or rNF_L is solved by Lemma 7.3. So, we assume that G_1 is an L-resolvent of $G\theta$ and ϕ_1 using θ_1. This means that $i_1 = 1$. Let $\phi_1 = \Box^s(A \leftarrow B_1, \ldots, B_m)$, $G = \leftarrow \alpha_1, \ldots, \alpha_k$, and the atom corresponding to the selected atom of $G\theta$ be $\alpha_i = \triangle A'$, where $A'\theta$ is the atom being unified with the forward labeled form of A using θ_1. Now $\theta\theta_1$ is a unifier for A' and the forward labeled form of A. The result of resolving G and ϕ_1 using $\theta\theta_1$ is exactly G_1. Thus we obtained an unrestricted refutation of $P \cup \{G\}$ in L, which looks exactly like the given refutation of $P \cup \{G\theta\}$, except the original goal is different and the first unifier is $\theta\theta_1$. Now apply the mgu lemma.

Proof of the lemma 7.11
Suppose α has variables $x_1, \ldots, x_n$. Let $a_1, \ldots, a_n$ be distinct constants not appearing in P and α, and let θ be the substitution $\{x_1/a_1, \ldots, x_n/a_n\}$. Then it is clear that $\alpha\theta$ is a logical consequence in L of P. Since $\alpha\theta$ is ground, by Lemma 7.10, $P \cup \{\leftarrow \alpha\theta\}$ has a refutation in L. Since the a_i do not appear in P or α, by replacing a_i by x_i (for $1 \leq i \leq n$) in this refutation, we obtain a refutation of $P \cup \{\leftarrow \alpha\}$ in L with the identity substitution as the computed answer.

Proof of the completeness theorem 7.12
Suppose G is the goal $\leftarrow \alpha_1, \ldots, \alpha_k$. Since θ is correct, $\forall((\alpha_1 \wedge \ldots \wedge \alpha_k)\theta)$ is a logical consequence of P in L. By Lemma 7.11, there exists a refutation of $P \cup \{\alpha_i\theta\}$ in L such that the computed answer is the identity substitution, for $1 \leq i \leq k$. We can combine these refutations into a refutation of $P \cup \{G\theta\}$ such that the computed answer is the identity substitution. Change each variable x occurring in the input clauses of the refutation of $P \cup \{G\theta\}$ by x' that does not occur in G and θ. Then recompute the refutation in the way such that when unifying $\{x, x'\}$ the substitution $\{x'/x\}$ (x' is changed to x) is used instead of $\{x/x'\}$. The resulting derivation will still be an SLD-refutation for $P \cup \{G\theta\}$ with the identity substitution as the computed answer. Applying the lifting lemma, we conclude that there exists a refutation of $P \cup \{G\}$ in L with computed answer γ such that $G\theta = G\gamma\delta$, for some substitution δ.