An SLD-Resolution Calculus for Basic Serial Multimodal Logics

Linh Anh Nguyen

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

Abstract. We develop semantics for modal logic programs in basic serial multimodal logics, which are parameterized by an arbitrary combination of generalized versions of axioms T, B, 4, 5 (in the form, e.g., $4 : \Box_i\varphi \rightarrow \Box_j\Box_k\varphi$) and $I : \Box_i\varphi \rightarrow \Box_j\varphi$. We do not assume any special restriction for the form of programs and goals. Our fixpoint semantics and SLD-resolution calculus are defined using the direct approach and closely reflect the axioms of the used modal logic. We prove that our SLD-resolution calculus is sound and complete.

1 Introduction

Classical logic programming is very useful in practice and has been thoroughly studied by many researchers. There are three standard semantics for definite logic programs: the least model semantics, the fixpoint semantics, and the SLD-resolution calculus (a procedural semantics) [9]. SLD-resolution was first described by Kowalski [8] for logic programming. It is a top-down procedure for answering queries in definite logic programs. On the other hand, the fixpoint semantics of logic programs is a bottom-up method for answering queries and was first introduced by van Emden and Kowalski [16] using the direct consequence operator T_P. This operator is monotonic, continuous, and has the least fixpoint $T_P \uparrow \omega = \bigcup_{n=0}^{\omega} T_P \uparrow n$, which forms the least Herbrand model of the given logic program P.

Multimodal logics are useful in many areas of computer science. For example, multimodal logics are used in knowledge representation and multi-agent systems by interpreting $\Box_i\varphi$ as "agent i knows/believes that φ is true". Modal extensions have been proposed for logic programming. There are two approaches to modal logic programming: the direct approach [6,1,2,10,13] and the translational approach [4,14]. The first approach directly uses modalities, while the second one translates modal logic programs to classical logic programs.

In [4], Debart *et al.* applied a functional translation technique for logic programs in multimodal logics which 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 $\Box_i\varphi \rightarrow \Box_j\varphi$. The technique is similar to the one used in Ohlbach's resolution calculus for modal logics [15]. Extra parameters are added to predicate symbols to represent paths in the Kripke model, and special unification

D.V. Hung and M. Wirsing (Eds.): ICTAC 2005, LNCS 3722, pp. 151–165, 2005.

algorithms are used to deal with them. In [14], Nonnengart proposed a semi-functional translation. His approach uses accessibility relations for translated programs, but with optimized clauses for representing properties of the accessibility relations, and does not modify unification. Nonnengart [14] applied the approach for modal logic programs in basic serial monomodal logics. He also gave an example in a multimodal logic of type $KD45$.

The translational approach seems attractive: just translate and it is done. However, the problem is more complicated. Using modal logics adds more nondeterminism to the search process, which cannot be eliminated but must be dealt with in some way. In the functional translation [4], the modified unification algorithm may return many substitutions, which causes branching. In the semi-functional translation [14], additional nondeterminism is caused by clauses representing frame restrictions of the used modal logic. In the direct approach considered shortly, additional nondeterminism is caused by modal rules which are used as meta clauses. Our point of view is that the direct approach is justifiable, as it deals with modalities more closely and "modalities allow us to separate object-level and epistemic-level notions nicely".

Using the direct approach for modal logic programming, Balbiani *et al.* [1] gave a declarative semantics and an SLD-resolution calculus for a class of logic programs in the monomodal logics KD, T, and $S4$. The work assumes that the modal operator $\Box$ does not occur in bodies of program clauses and goals. In [2], Baldoni *et al.* gave a framework for developing declarative and operational semantics for logic programs in multimodal logics which have axioms of the form $[t_1] \dots [t_n]\varphi \rightarrow [s_1] \dots [s_m]\varphi$, where $[t_i]$ and $[s_j]$ are universal modal operators indexed by terms t_i and s_j, respectively. In that work, existential modal operators are disallowed in programs and goals.

In [10], we developed a fixpoint semantics, the least model semantics, and an SLD-resolution calculus in a direct way for modal logic programs in all of the basic serial monomodal logics KD, T, KDB, B, $KD4$, $S4$, $KD5$, $KD45$, and $S5$. We also extended the SLD-resolution calculus for the almost serial monomodal logics KB, $K5$, $K45$, and $KB5$. There are two important properties of our approach in [10]: no special restriction on occurrences of $\Box$ and $\Diamond$ is assumed and the semantics are formulated closely to the style of classical logic programming (as in Lloyd's book [9]).

The aim of this paper is to generalize the methods and results of our above-mentioned work for the whole class of *basic serial multimodal logics* (BSMM). A BSMM logic is an extension of the multimodal logic K_m with the axioms of seriality $D : \Box_i\varphi \rightarrow \Diamond_i\varphi$ and any combination of axioms of the form $T : \Box_i\varphi \rightarrow \varphi$ or $I : \Box_i\varphi \rightarrow \Box_j\varphi$ or $\varphi \rightarrow \Box_i\Diamond_j\varphi$ or $\Box_i\varphi \rightarrow \Box_j\Box_k\varphi$ or $\Diamond_i\varphi \rightarrow \Box_j\Diamond_k\varphi$, where i, j, k can be arbitrary or related somehow. Note that the last three schemata are generalized versions of the axioms B, 4, and 5, respectively.

Using the framework presented in our manuscript [13], we give a fixpoint semantics, the least model semantics, and an SLD-resolution calculus for modal logic programs in any BSMM logic. We prove that the calculus is sound and

complete. Due to the lack of space, we do not present proofs involving with the fixpoint semantics and the least model semantics.

From the view of SLD-resolution, our idea is to use labeled existential modal operators to break a complex goal into simple goal atoms and to use modal axioms as meta clauses. For example, we cannot break $\leftarrow \Diamond_i(A \wedge B)$ into $\leftarrow \Diamond_i A, \Diamond_i B$, but if we label the operator $\Diamond_i$ by X then we can safely break $\leftarrow \langle X\rangle_i(A \wedge B)$ into $\leftarrow \langle X\rangle_i A, \langle X\rangle_i B$. Additionally, for example, we use the axiom $\Diamond_i\varphi \rightarrow \Box_j\Diamond_k\varphi$ (and their reverse $\Diamond_j\Box_k\varphi \rightarrow \Box_i\varphi$) in the form of meta clauses $\triangle\Box_j\Diamond_k\triangle' E \leftarrow \triangle\Diamond_i\triangle' E$ and $\triangle\Box_i\triangle' E \leftarrow \triangle\Diamond_j\Box_k\triangle' E$, where $\triangle$ and $\triangle'$ are sequences of modal operators and E is a classical atom.

2 Preliminaries

2.1 Syntax and Semantics of Quantified Multimodal Logics

A language for quantified multimodal logics is an extension of the language of classical predicate logic with modal operators $\Box_i$ and $\Diamond_i$, for $1 \leq i \leq m$ (where m is fixed). The modal operators $\Box_i$ and $\Diamond_i$ can take various meanings. For example, $\Box_i$ can stand for "the agent i believes" and $\Diamond_i$ for "it is considered possible by agent i". The operators $\Box_i$ are called universal modal operators, while $\Diamond_i$ are called existential modal operators. Terms and formulas are defined in the usual way, with the addition that if φ is a formula then $\Box_i\varphi$ and $\Diamond_i\varphi$ are also formulas.

A *Kripke frame* is a tuple $\langle W, \tau, R_1, \ldots, R_m\rangle$, where W is a nonempty set of possible worlds, $\tau \in W$ is the *actual world*, and R_i for $1 \leq i \leq m$ is a binary relation on W, called the *accessibility relation* for the modal operators $\Box_i$, $\Diamond_i$. If $R_i(w, u)$ holds then we say that u is accessible from w via R_i.

A *fixed-domain Kripke model with rigid terms*, hereafter simply called a Kripke model or just a model, is a tuple $M = \langle D, W, \tau, R_1, \ldots, R_m, \pi\rangle$, where D is a set called the *domain*, $\langle W, \tau, R_1, \ldots, R_m\rangle$ is a Kripke frame, and π is an interpretation of constant symbols, function symbols and predicate symbols. For a constant symbol a, $\pi(a)$ is an element of D, denoted by a^M. For an n-ary function symbol f, $\pi(f)$ is a function from D^n to D, denoted by f^M. For an n-ary predicate symbol p and a world $w \in W$, $\pi(w)(p)$ is an n-ary relation on D, denoted by $p^{M,w}$.

A *model graph* is a tuple $\langle W, \tau, R_1, \ldots, R_m, H\rangle$, where $\langle W, \tau, R_1, \ldots, R_m\rangle$ is a Kripke frame and H is a function that maps each world of W to a set of formulas.

Every model graph $\langle W, \tau, R_1, \ldots, R_m, H\rangle$ corresponds to a Herbrand model $M = \langle \mathcal{U}, W, \tau, R_1, \ldots, R_m, \pi\rangle$ specified by: $\mathcal{U}$ is the Herbrand universe (i.e. the set of all ground terms), $c^M = c$, $f^M(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$, and $((t_1, \ldots, t_n) \in p^{M,w}) \equiv (p(t_1, \ldots, t_n) \in H(w))$, where $t_1, \ldots, t_n$ are ground terms. We will sometimes treat a model graph as its corresponding model.

A *variable assignment* V w.r.t. a Kripke model M is a function that maps each variable to an element of the domain of M. The value of $t^M[V]$ for a term t is defined as usual.

Given some Kripke model $M = \langle D, W, \tau, R_1, \ldots, R_m, \pi \rangle$, some variable assignment V, and some world $w \in W$, the *satisfaction relation* $M, V, w \vDash \psi$ for a formula ψ is defined as follows:

$$\begin{array}{lll} M, V, w \vDash p(t_1, \ldots, t_n) & \text{iff} & (t_1^M[V], \ldots, t_n^M[V]) \in p^{M,w}; \\ M, V, w \vDash \Box_i \varphi & \text{iff} & \text{for all } v \in W \text{ such that } R_i(w, v),\ M, V, v \vDash \varphi; \\ M, V, w \vDash \forall x.\varphi & \text{iff} & \text{for all } a \in D,\ (M, V', w \vDash \varphi), \\ & & \quad \text{where } V'(x) = a \text{ and } V'(y) = V(y) \text{ for } y \neq x; \end{array}$$

and as usual for the other cases (treating $\Diamond_i \varphi$ as $\neg\Box_i\neg\varphi$, and $\exists x.\varphi$ as $\neg\forall x.\neg\varphi$). We say that M satisfies φ, or φ is true in M, and write $M \vDash \varphi$, if $M, V, \tau \vDash \varphi$ for every V. For a set Γ of formulas, we call M a model of Γ and write $M \vDash \Gamma$ if $M \vDash \varphi$ for every $\varphi \in \Gamma$.

If the class of admissible interpretations contains all Kripke models (with no restrictions on the accessibility relations) then we have a quantified multimodal logic which has a standard Hilbert-style axiomatization denoted by K_m. Other *normal (multi)modal logics* are obtained by adding certain axioms to K_m. Mostly used axioms are ones that correspond to a certain restriction on the Kripke frame defined by a classical first-order formula using the accessibility relations. For example, the axiom $(D) : \Box_i \varphi \rightarrow \Diamond_i \varphi$ corresponds to the frame restriction $\forall x\, \exists y\, R_i(x, y)$. Normal modal logics containing this axiom (for all $1 \leq i \leq m$) are called *serial* modal logics.

For a normal modal logic L whose class of admissible interpretations can be characterized by classical first-order formulas of the accessibility relations, we call such formulas *L-frame restrictions*, and call frames with such properties *L-frames*. We call a model M with an 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 formulas, we write $\Gamma \vDash_L \varphi$ and call φ a *logical consequence* of Γ in L if φ is true in every L-model of Γ.

2.2 Basic Serial Multimodal Logics

A normal multimodal logic can be characterized by axioms extending the system K_m. Consider the class $BSMM$ of basic serial multimodal logics specified as follows. A $BSMM$ logic is a normal multimodal logic parameterized by relations $AD/1$, $AT/1$, $AI/2$, $AB/2$, $A4/3$, $A5/3$ on the set $\{1, \ldots, m\}$, where the numbers on the right are arities and AD is required to be full. These relations specify the following axioms:

$$\begin{array}{ll} \Box_i \varphi \rightarrow \Diamond_i \varphi & \text{if } AD(i) \\ \Box_i \varphi \rightarrow \varphi & \text{if } AT(i) \\ \Box_i \varphi \rightarrow \Box_j \varphi & \text{if } AI(i, j) \\ \varphi \rightarrow \Box_i \Diamond_j \varphi & \text{if } AB(i, j) \\ \Box_i \varphi \rightarrow \Box_j \Box_k \varphi & \text{if } A4(i, j, k) \\ \Diamond_i \varphi \rightarrow \Box_j \Diamond_k \varphi & \text{if } A5(i, j, k) \end{array}$$

It can be shown that the above axioms correspond to the following frame restrictions in the sense that by adding some of the axioms to the system K_m

we obtain an axiomatization system which is sound and complete with respect to the class of admissible interpretations that satisfy the corresponding frame restrictions.

Axiom	Corresponding Condition
$\Box_i\varphi \rightarrow \Diamond_i\varphi$	$\forall u\ \exists v\ R_i(u,v)$
$\Box_i\varphi \rightarrow \varphi$	$\forall u\ R_i(u,u)$
$\Box_i\varphi \rightarrow \Box_j\varphi$	$R_j \subseteq R_i$
$\varphi \rightarrow \Box_i\Diamond_j\varphi$	$\forall u,v\ (R_i(u,v) \rightarrow R_j(v,u))$
$\Box_i\varphi \rightarrow \Box_j\Box_k\varphi$	$\forall u,v,w\ (R_j(u,v) \wedge R_k(v,w) \rightarrow R_i(u,w))$
$\Diamond_i\varphi \rightarrow \Box_j\Diamond_k\varphi$	$\forall u,v,w\ (R_i(u,v) \wedge R_j(u,w) \rightarrow R_k(w,v))$

For a $BSMM$ logic L, we define the set of L-frame restrictions to be the set of the frame restrictions corresponding to the tuples of the relations AD, AT, AI, AB, $A4$, $A5$. We also use $BSMM$ to denote an arbitrary logic belonging to the $BSMM$ class.

For further reading on first-order modal logic, see, e.g., [3,7].

2.3 Modal Logic Programs

A *modality* is a sequence of modal operators, which may be empty. A *universal modality* is a modality containing only universal modal operators. We use $\triangle$ to denote a modality and $\boxdot$ to denote a universal modality. Similarly as in classical logic programming, we use the clausal form $\boxdot(\varphi \leftarrow \psi_1, \ldots, \psi_n)$ to denote the formula $\forall(\boxdot(\varphi \vee \neg\psi_1 \ldots \vee \neg\psi_n))$. We use E to denote a classical atom.

A *program clause* is a formula of the form $\boxdot(A \leftarrow B_1, \ldots, B_n)$, where $n \geq 0$ and $A, B_1, \ldots, B_n$ are formulas of the form E, $\Box_i E$, or $\Diamond_i E$. $\boxdot$ is called the *modal context*, A the *head*, and $B_1, \ldots, B_n$ the *body* of the program clause. An *MProlog program* is a finite set of program clauses.

An *MProlog goal atom* is a formula of the form $\boxdot E$ or $\boxdot\Diamond_i E$. An *MProlog goal* is a formula written in the clausal form $\leftarrow \alpha_1, \ldots, \alpha_k$, where each α_i is an MProlog goal atom. We denote the *empty goal* (the *empty clause*) by $\diamond$.

Let P be an MProlog program and $G = \leftarrow \alpha_1, \ldots, \alpha_k$ be an MProlog goal. An *answer* θ for $P \cup \{G\}$ is a substitution whose domain is the set of all variables of G. We say that θ is a *correct answer* in L for $P \cup \{G\}$ if θ is an answer for $P \cup \{G\}$ and $P \vDash_L \forall((\alpha_1 \wedge \ldots \wedge \alpha_k)\theta)$.

It is shown in [11] that MProlog has the same expressiveness power as the general Horn fragment in normal modal logics.

3 Semantics of MProlog Programs in BSMM

In this section, we present a fixpoint semantics, the least model semantics, and an SLD-resolution calculus for MProlog programs in a BSMM logic L.

3.1 Labeled Modal Operators

When applying the direct consequence operator $T_{L,P}$ for an MProlog program P in L, if we obtain an "atom" of the form $\triangle\Diamond_i E$, where $\triangle$ is a sequence

of modal operators, then to simplify the task we label the modal operator $\Diamond_i$. Labeling allows us to address the chosen world(s) in which this particular E must hold. A natural way is to label $\Diamond_i$ by E to obtain $\langle E\rangle_i$. On the other hand, when dealing with SLD-derivation, we cannot change a goal $\leftarrow \Diamond_i(A \wedge B)$ to $\leftarrow \Diamond_i A, \Diamond_i B$. But if we label the operator $\Diamond_i$, let's say by X, then we can safely change $\leftarrow \langle X\rangle_i(A \wedge B)$ to $\leftarrow \langle X\rangle_i A, \langle X\rangle_i B$.

We will use the following notations:

- $\top$: the *truth* symbol, with the usual semantics;
- E, F : classical atoms (which may contain variables) or $\top$;
- X, Y, Z : variables for classical atoms or $\top$, called *atom variables*;
- $\langle E\rangle_i$, $\langle X\rangle_i$: $\Diamond_i$ labeled by E or X;
- ∇ : $\Box_i$, $\Diamond_i$, $\langle E\rangle_i$, or $\langle X\rangle_i$, called a modal operator;
- $\triangle$: a (possibly empty) sequence of modal operators, called a *modality*;
- $\boxdot$: a *universal modality*;
- A, B : formulas of the form E or ∇E, called *simple atoms*;
- α, β : formulas of the form $\triangle E$, called *atoms*;
- φ, ψ : *(labeled) formulas* (i.e. formulas that may contain $\langle E\rangle_i$ and $\langle X\rangle_i$).

We use subscripts beside ∇ to indicate modal indexes in the same way as for $\Box$ and $\Diamond$. To distinguish a number of modal operators we use superscripts of the form (i), e.g. $\Box^{(1)}$, $\Box^{(2)}$, $\nabla^{(i)}$, $\nabla^{(i')}$.

A *ground formula* is a formula with no variables and no atom variables. A modal operator is said to be *ground* if it is $\Box_i$, $\Diamond_i$, or $\langle E\rangle_i$ with E being $\top$ or a ground classical atom. A *ground modality* is a modality that contains only ground modal operators. A *labeled modal operator* is a modal operator of the form $\langle E\rangle_i$ or $\langle X\rangle_i$.

Denote $\mathit{EdgeLabels} = \{\langle E\rangle_i \mid E \in \mathcal{B} \cup \{\top\} \text{ and } 1 \leq i \leq m\}$, where $\mathcal{B}$ is the Herbrand base (i.e. the set of all ground classical atoms). The semantics of $\langle E\rangle_i \in \mathit{EdgeLabels}$ is specified as follows. Let $M = \langle D, W, \tau, R_1, \ldots, R_m, \pi\rangle$ be a Kripke model. A *$\Diamond$-realization function on M* is a partial function $\sigma : W \times \mathit{EdgeLabels} \rightarrow W$ such that if $\sigma(w, \langle E\rangle_i) = u$, then $R_i(w, u)$ holds and $M, u \vDash E$. Given a $\Diamond$-realization function σ, a world $w \in W$, and a ground formula φ, the satisfaction relation $M, \sigma, w \vDash \varphi$ is defined in the usual way, except that $M, \sigma, w \vDash \langle E\rangle_i\psi$ iff $\sigma(w, \langle E\rangle_i)$ is defined and $M, \sigma, \sigma(w, \langle E\rangle_i) \vDash \psi$. We write $M, \sigma \vDash \varphi$ to denote that $M, \sigma, \tau \vDash \varphi$. For a set I of ground atoms, 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.

3.2 Model Generators

A modality is in *labeled form* if it does not contain modal operators of the form $\Diamond_i$ or $\langle\top\rangle_i$. An atom is in *labeled form* (resp. *almost labeled form*) if it is of the form $\triangle E$ (resp. $\triangle A$) with $\triangle$ in labeled form.

A *model generator* is a set of ground atoms not containing $\Diamond_i$, $\langle\top\rangle_i$, $\top$.

We will define the *standard L-model* of a model generator I so that it is a *least L-model* of I (where a model M is *less than or equal to* a model M' if

Table 1. A schema for semantics of MProlog in $BSMM$

$L = BSMM$	
Rules specifying Ext_L and Sat_L:	
$\triangle\langle E\rangle_i\alpha \rightarrow \triangle\Diamond_i\alpha$	(1)
$\triangle\Box_i\alpha \rightarrow \triangle\Diamond_i\alpha$	(2)
$\triangle\Box_i\alpha \rightarrow \triangle\alpha$ if $AT(i)$	(3)
$\triangle\alpha \rightarrow \triangle\Diamond_i\alpha$ if $AT(i)$	(4)
$\triangle\Box_i\alpha \rightarrow \triangle\Box_j\alpha$ if $AI(i,j)$	(5)
$\triangle\Diamond_j\alpha \rightarrow \triangle\Diamond_i\alpha$ if $AI(i,j)$	(6)
$\triangle\alpha \rightarrow \triangle\Box_i\Diamond_j\alpha$ if $AB(i,j)$	(7)
$\triangle\Diamond_i\Box_j\alpha \rightarrow \triangle\alpha$ if $AB(i,j)$	(8)
$\triangle\Box_i\alpha \rightarrow \triangle\Box_j\Box_k\alpha$ if $A4(i,j,k)$	(9)
$\triangle\Diamond_j\Diamond_k\alpha \rightarrow \triangle\Diamond_i\alpha$ if $A4(i,j,k)$	(10)
$\triangle\Diamond_i\alpha \rightarrow \triangle\Box_j\Diamond_k\alpha$ if $A5(i,j,k)$	(11)
$\triangle\Diamond_j\Box_k\alpha \rightarrow \triangle\Box_i\alpha$ if $A5(i,j,k)$	(12)
Rules specifying $rSat_L$:	
$\triangle\Diamond_i\alpha \leftarrow \triangle\langle X\rangle_i\alpha$ where X is a fresh atom variable	(1)
$\triangle\nabla_i\alpha \leftarrow \triangle\Box_i\alpha$	(2)
plus a rule $\alpha \leftarrow \beta$ for each k-th rule $\beta \rightarrow \alpha$ specifying Sat_L, $k \geq 3$, with the same accompanying condition	(3)..(12)
Comments w.r.t. [13]:	
$\preceq_L$ is denoted by $\preceq$ and defined in page 158.	
No restrictions on L-normal form of modalities.	
No rules specifying NF_L and rNF_L.	

for every positive ground formula φ without labeled operators, if $M \models \varphi$ then $M' \models \varphi$). In the construction we will use the operator Ext_L defined below.

A *forward rule* is a schema of the form $\alpha \rightarrow \beta$, while a *backward rule* is a schema of the form $\alpha \leftarrow \beta$. A rule can be accompanied with some conditions specifying when the rule can be applied.

The *operator* Ext_L is specified by the corresponding forward rules given in Table 3.2. Given a model generator I, $Ext_L(I)$ is the least extension of I that contains all ground atoms in labeled form that are derivable from some atom of I using the rules specifying Ext_L.

Define $Serial = \{\boxdot\langle\top\rangle_i\top \mid 1 \leq i \leq m\}$.

Let I be a model generator. The *standard L-model* of I is defined as follows. Let $W' = EdgeLabels^*$ (i.e. the set of finite sequences of elements of $\{\langle E\rangle_i \mid E \in \mathcal{B} \cup \{\top\}$ and $1 \leq i \leq m\}$), $\tau = \epsilon$, $H(\tau) = Ext_L(I) \cup Serial$. Let $R'_i \subseteq W' \times W'$ and $H(u)$, for $u \in W'$, $u \neq \tau$, be the least sets such that:

- if $\langle E\rangle_i\alpha \in H(w)$, then $R'_i(w, w\langle E\rangle_i)$ holds and $\{E, \alpha\} \subseteq H(w\langle E\rangle_i)$;
- if $\Box_i\alpha \in H(w)$ and $R'_i(w, w\langle E\rangle_i)$ holds, then $\alpha \in H(w\langle E\rangle_i)$.

Let R_i, for $1 \leq i \leq m$, be the least extension of R'_i such that $\{R_i \mid 1 \leq i \leq m\}$ satisfies all the L-frame restrictions except seriality (which is cared by $Serial$)[1]. Let W be W' without worlds not accessible directly nor indirectly from τ via the accessibility relations R_i. We call the model graph $\langle W, \tau, R_1, \ldots, R_m, H\rangle$ the *standard L-model graph* of I, and its corresponding model M the *standard L-model* of I. $\{R'_i \mid 1 \leq i \leq m\}$ is called the *skeleton* of M. By the *standard $\Diamond$-realization function on M* we call the $\Diamond$-realization function σ defined as follows: if $R'_i(w, w\langle E\rangle_i)$ holds then $\sigma(w, \langle E\rangle_i) = w\langle E\rangle_i$, else $\sigma(w, \langle E\rangle_i)$ is undefined.

It is shown in [11] that *the standard L-model of a model generator I is a least L-model of I*.

3.3 Fixpoint Semantics

We now consider the direct consequence operator $T_{L,P}$. Given a model generator I, how can $T_{L,P}(I)$ be defined? Basing on the axioms of L, I is first extended to the *L-saturation* of I, denoted by $Sat_L(I)$, which is a set of atoms. Next, *L-instances of program clauses* of P are *applied* to the atoms of $Sat_L(I)$. This is done by the operator $T_{0L,P}$. Then $T_{L,P}(I)$ is defined as $T_{0L,P}(Sat_L(I))$.

To compare modal operators we define $\preceq$ to be the least reflexive and transitive binary relation between modal operators such that $\Diamond_i \preceq \langle E\rangle_i \preceq \Box_i$ and $\Diamond_i \preceq \langle X\rangle_i \preceq \Box_i$.

An atom $\nabla^{(1)} \ldots \nabla^{(n)}\alpha$ is called an *instance* of an atom $\nabla^{(1')} \ldots \nabla^{(n')}\alpha'$ if there exists a substitution θ such that $\alpha = \alpha'\theta$ and $\nabla^{(i)} \preceq \nabla^{(i')}\theta$ for all $1 \leq i \leq n$ (treating $\nabla^{(i')}$ as an expression). For example, $\langle X\rangle_1\Diamond_2 E$ is an instance of $\Box_1\langle F\rangle_2 E$.

A modality $\triangle$ is called an *instance* of $\triangle'$, and we also say that $\triangle'$ is *equal to* or *more general in L 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.

Let $\boxdot$ and $\boxdot'$ be universal modalities. We say that $\boxdot$ is an *L-context instance* of $\boxdot'$ if $\boxdot'\varphi \rightarrow \boxdot\psi$ is L-valid (for every ψ). This is defined semantically, and in general, the problem of checking whether $\boxdot$ is an L-context instance of $\boxdot'$ for an *input* BSMM logic L is perhaps undecidable. However, the problem is decidable for many modal logics, including basic monomodal logics, multimodal logics of belief [11], and regular grammar logics [5].

Let φ and φ' be program clauses with empty modal context. We say that $\boxdot\varphi$ is an *L-instance* of (a program clause) $\boxdot'\varphi'$ if $\boxdot$ is an L-context instance of $\boxdot'$ and there exists a substitution θ such that $\varphi = \varphi'\theta$.

We now give definitions for Sat_L and $T_{0L,P}$.

The *saturation operator* Sat_L is specified by the corresponding forward rules given in Table 3.2. Given a model generator I, $Sat_L(I)$ is the least extension of I that contains all ground atoms in *almost* labeled form that are derivable from some atom in I using the rules specifying Sat_L. (Note that the rules specifying Sat_L are the same as the rules specifying Ext_L, but these operators are different.)

[1] The least extension exists due to the assumption that all L-frame restrictions not concerning seriality are classical first-order Horn formulas.

When computing the least fixpoint of a modal logic program, whenever an atom of the form $\triangle\Diamond_i E$ is introduced, we "fix" the $\Diamond$ by replacing the atom by $\triangle\langle E\rangle_i E$. This leads to the following definition. The *forward labeled form* of an atom α is the atom α' such that if α is of the form $\triangle\Diamond_i E$ then $\alpha' = \triangle\langle E\rangle_i E$, else $\alpha' = \alpha$.

Let P be an L-MProlog program. The *operator* $T_{^0L,P}$ is defined as follows: for a set I of ground atoms in almost labeled form, $T_{^0L,P}(I)$ is the least (w.r.t. $\subseteq$) model generator such that if $\boxdot(A \leftarrow B_1, \ldots, B_n)$ is a ground L-instance of some program clause of P and $\triangle$ is a maximally general[2] ground modality *in labeled form* such that $\triangle$ is an L-instance of $\boxdot$ and $\triangle B_i$ is an instance of some atom of I for every $1 \leq i \leq n$, then the forward labeled form of $\triangle A$ belongs to $T_{^0L,P}(I)$.

Define $T_{L,P}(I) = T_{^0L,P}(Sat_L(I))$. By definition, the operators Sat_L and $T_{^0L,P}$ are both increasingly monotonic and compact. Hence the operator $T_{L,P}$ is monotonic and continuous. By the Kleene theorem, it follows that $T_{L,P}$ has the least fixpoint $T_{L,P}\uparrow\omega = \bigcup_{n=0}^{\omega} T_{L,P}\uparrow n$, where $T_{L,P}\uparrow 0 = \emptyset$ and $T_{L,P}\uparrow n = T_{L,P}(T_{L,P}\uparrow(n-1))$ for $n > 0$.

Denote the least fixpoint $T_{L,P}\uparrow\omega$ by $I_{L,P}$ and the standard L-model of $I_{L,P}$ by $M_{L,P}$. It is shown in [11] that $M_{L,P}$ *is a least* L*-model of* P.

Example 1. Consider the multimodal logic L specified by $m = 2$, $AD = \{1, 2\}$, $AT = \{1\}$, $AI = \{(2, 1)\}$, and $AB = A4 = A5 = \emptyset$. In other words, the logic is characterized by the axioms: $\Box_1\varphi \rightarrow \Diamond_1\varphi$; $\Box_2\varphi \rightarrow \Diamond_2\varphi$; $\Box_1\varphi \rightarrow \varphi$; and $\Box_2\varphi \rightarrow \Box_1\varphi$. Consider the following program P:

$$\begin{aligned}\varphi_1 &= \Diamond_2 p(a) \leftarrow \\ \varphi_2 &= \Box_2(\Box_1 q(x) \leftarrow \Diamond_2 p(x)) \\ \varphi_3 &= \Box_2(r(x) \leftarrow p(x), q(x))\end{aligned}$$

We have $T_{L,P}\uparrow 1 = \{\langle p(a)\rangle_2 p(a)\}$ and

$$Sat_L(T_{L,P}\uparrow 1) = \{\langle p(a)\rangle_2 p(a), \langle p(a)\rangle_2\Diamond_1 p(a), \langle p(a)\rangle_2\Diamond_2 p(a)\}$$

The program clause φ_2 has two L-instances applicable to $Sat_L(T_{L,P} \uparrow 1)$: the clause φ_2 itself and $\Box_1 q(x) \leftarrow \Diamond_2 p(x)$. Applying these clauses to $Sat_L(T_{L,P}\uparrow 1)$, we obtain $T_{L,P}\uparrow 2 = T_{L,P}\uparrow 1 \cup \{\langle p(a)\rangle_2\Box_1 q(a),\ \Box_1 q(a)\}$. Observe that the set $Sat_L(T_{L,P}\uparrow 2)$ contain both $\langle p(a)\rangle_2 p(a)$ and $\langle p(a)\rangle_2 q(a)$. Hence, by applying the program clause φ_3, we have $\langle p(a)\rangle_2 r(a) \in T_{L,P}\uparrow 3$ and arrive at

$$I_{L,P} = T_{L,P}\uparrow 3 = \{\langle p(a)\rangle_2 p(a), \langle p(a)\rangle_2\Box_1 q(a), \Box_1 q(a), \langle p(a)\rangle_2 r(a)\}$$

3.4 SLD-Resolution

The main work in developing an SLD-resolution calculus for MProlog in L is to specify a reverse analogue of the operator $T_{L,P}$. The operator $T_{L,P}$ is a composition of Sat_L and $T_{^0L,P}$. So, we have to investigate reversion of these operators.

[2] W.r.t. the pre-order between modalities described earlier.

A *goal* is a clause of the form $\leftarrow \alpha_1, \ldots, \alpha_k$, where each α_i is an atom.

The following definition concerns reversion of the operator $T_{0L,P}$.

Let $G = \leftarrow \alpha_1, \ldots, \alpha_i, \ldots, \alpha_k$ be a goal and $\varphi = \boxdot(A \leftarrow B_1, \ldots, B_n)$ a program clause. Then G' is *derived* from G and φ in L using mgu θ, and called an *L-resolvent* of G and φ, if the following conditions hold:

- $\alpha_i = \triangle' A'$, with $\triangle'$ *in labeled form*, is called the *selected atom*, and A' is called the *selected head atom*;
- $\triangle'$ is an instance of a universal modality $\boxdot'$ and $\boxdot'(A \leftarrow B_1, \ldots, B_n)$ is an L-instance of the program clause φ;
- θ is an mgu of A' and the forward labeled form of A;
- G' is the goal $\leftarrow (\alpha_1, \ldots, \alpha_{i-1}, \triangle' B_1, \ldots, \triangle' B_n, \alpha_{i+1}, \ldots, \alpha_k)\theta$.

As a reverse analogue of the operator Sat_L, we provide the operator $rSat_L$, which is specified by the corresponding backward rules given in Table 3.2. We say that $\beta = rSat_L(\alpha)$ *using an* $rSat_L$ *rule* $\alpha' \leftarrow \beta'$ if $\alpha \leftarrow \beta$ is of the form $\alpha' \leftarrow \beta'$. We write $\beta = rSat_L(\alpha)$ to denote that "$\beta = rSat_L(\alpha)$ using some $rSat_L$ rule".

Let $G = \leftarrow \alpha_1, \ldots, \alpha_i, \ldots, \alpha_k$ be a goal. If $\alpha_i' = rSat_L(\alpha_i)$ using an $rSat_L$ rule φ, then $G' = \leftarrow \alpha_1, \ldots, \alpha_{i-1}, \alpha_i', \alpha_{i+1}, \ldots, \alpha_k$ is *derived* from G and φ, and we call G' an *(L-)resolvent* of G and φ, and α_i the *selected atom* of G.

Observe that $rSat_L$ rules are similar to program clauses and the way of applying them is similar to the way of applying classical program clauses, except that we do not need mgu's.

We now define SLD-derivation and SLD-refutation.

Let P be an MProlog program and G a goal. An *SLD-derivation* from $P \cup \{G\}$ in L consists of a (finite or infinite) sequence $G_0 = G, G_1, \ldots$ of goals, a sequence $\varphi_1, \varphi_2, \ldots$ of variants of program clauses of P or $rSat_L$ rules, and a sequence $\theta_1, \theta_2, \ldots$ of mgu's such that if φ_i is a variant of a program clause then G_i is derived from G_{i-1} and φ_i in L using θ_i, else $\theta_i = \varepsilon$ (the empty substitution) and G_i is derived from G_{i-1} and (the $rSat_L$ rule variant) φ_i. Each φ_i is called an *input clause/rule* of the derivation.

We assume *standardizing variables apart* as usual (see [9]).

An *SLD-refutation* of $P \cup \{G\}$ in L is a finite SLD-derivation from $P \cup \{G\}$ in L which has the empty clause as the last goal in the derivation.

Let P be an MProlog program and G a goal. A *computed answer* θ in L of $P \cup \{G\}$ is the substitution obtained by restricting the composition $\theta_1 \ldots \theta_n$ to the variables of G, where $\theta_1, \ldots, \theta_n$ is the sequence of mgu's used in an SLD-refutation of $P \cup \{G\}$ in L.

Example 2. Reconsider the modal logic L and the program P given in Example 1. Let $G = \leftarrow \Diamond_2 r(x)$. We give below an SLD-refutation of $P \cup \{G\}$ in L with computed answer $\{x/a\}$.

Goals	Input clauses/rules	MGUs
$\leftarrow \Diamond_2 r(x)$		
$\leftarrow \langle X \rangle_2 r(x)$	(1)	

$\leftarrow \langle X\rangle_2 p(x), \langle X\rangle_2 q(x)$	φ_3	$\{x_2/x\}$
$\leftarrow \langle p(a)\rangle_2 q(a)$	φ_1	$\{X/p(a), x/a\}$
$\leftarrow \langle p(a)\rangle_2 \Box_1 q(a)$	(3)	
$\leftarrow \langle p(a)\rangle_2 \Diamond_2 p(a)$	φ_2	$\{x_5/a\}$
$\leftarrow \langle p(a)\rangle_2 \Diamond_1 p(a)$	(6)	
$\leftarrow \langle p(a)\rangle_2 p(a)$	(4)	
$\Diamond$	φ_1	

4 Soundness and Completeness of SLD-Resolution

In this section, we prove soundness and completeness of the SLD-resolution calculus given for MProlog in BSMM, which are stated as follows.

Theorem 1. *Let L be a BSMM logic, P an MProlog program, and G an MProlog goal. Then every computed answer in L of $P \cup \{G\}$ is a correct answer in L of $P \cup \{G\}$. Conversely, for every correct answer θ in L of $P \cup \{G\}$, there exists a computed answer γ in L of $P \cup \{G\}$ which is more general than θ (i.e. $\theta = \gamma\delta$ for some δ).*

4.1 How to Prove?

In [13], we presented a general framework for developing fixpoint semantics, the least model semantics, and SLD-resolution calculi for logic programs in multimodal logics and proved that under certain expected properties of a concrete instantiation of the framework for a specific multimodal logic, the SLD-resolution calculus is sound and complete. The semantics of MProlog in BSMM presented in the previous section and summarized in Table 3.2 are based on and compatible with the framework given in [13]. For $L = BSMM$, we have applied the following simplifications w.r.t. [13]:

- There are no restrictions on L-normal form of modalities and the normalization operator NF_L and it reverse rNF_L are just identity operators. The word *L-normal* is also omitted in "L-normal model generator".
- There are no restrictions on BSMM-MProlog, i.e. every MProlog program (resp. goal) is a BSMM-MProlog program (resp. goal).
- The index L is omitted in the notations $Serial_L$, $\preceq_L$, and "L-instance" (of an atom or a modality).

By the results of [13], to prove soundness and completeness of SLD-resolution for MProlog in BSMM, we can prove Expected Lemmas 4 – 10 of [13] (w.r.t. the schema given in Table 3.2). The Expected Lemma 6 is trivial. The Expected Lemma 10 and the part of Expected Lemma 8 involving with NF_L/rNF_L can be omitted because NF_L and rNF_L are identity operators. The Expected Lemmas 7 and 9 and the remaining part of Expected Lemma 8, which concern properties of the operators Sat_L and $rSat_L$, can be verified in a straightforward way. The remaining Expected Lemmas 4 and 5 are given below as Lemmas 1 and 2, respectively, and will be proved in this section.

A model generator I is called an *L-model generator of P* if $T_{L,P}(I) \subseteq I$.

Lemma 1. *Let P be an MProlog program and I an L-model generator of P. Then the standard L-model of I is an L-model of P.*

Lemma 2. *Let I be a model generator, M the standard L-model of I, and α a ground MProlog goal atom. Suppose that $M \vDash \alpha$. Then α is an instance of some atom of $Sat_L(I)$.*

4.2 Extended *L*-Model Graphs

To proceed we need extended L-model graphs and some properties of them. Let I be a model generator. Define Ext'_L to be the operator such that $Ext'_L(I)$ is the least set of atoms extending I and closed w.r.t. the rules specifying Ext_L. Note that we allow $Ext'_L(I)$ to contain atoms not in labeled form and have that $Ext_L(I) \subseteq Ext'_L(I)$. The *extended L-model graph* of I is defined in the same way as the standard L-model graph of I but with $Ext'_L(I)$ in the place of $Ext_L(I)$.

We need the two following auxiliary lemmas.

Lemma 3. *Let I be a model generator, M the standard L-model graph of I, and M' the extended L-model graph of I. Then M' has the same frame as M, and furthermore, if $M = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ and $M' = \langle W, \tau, R_1, \ldots, R_m, H'\rangle$ then for every $w \in W$, $H(w) \subseteq H'(w)$ and $H'(w) - H(w)$ is a set of formulas containing some unlabeled existential modal operators.*

The proof of this lemma is straightforward.

If a modality $\triangle$ is obtainable from $\triangle'$ by replacing some (possibly zero) ∇_i by $\Box_i$ then we call $\triangle$ a *$\Box$-lifting form* of $\triangle'$. If $\triangle$ is a $\Box$-lifting form of $\triangle'$ then we call an atom $\triangle\alpha$ a *$\Box$-lifting form* of $\triangle'\alpha$. For example, $\Box_1\langle p(a)\rangle_1\Box_2 q(b)$ is a $\Box$-lifting form of $\langle X\rangle_1\langle p(a)\rangle_1\Diamond_2 q(b)$.

Lemma 4. *Let I be a model generator and $M = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ be the extended L-model graph of I. Let $w = \langle E_1\rangle_{i_1} \ldots \langle E_k\rangle_{i_k}$ be a world of M and $\triangle = w$ be a modality. Then for α (resp. A) not containing $\top$, $\alpha \in H(w)$ (resp. $A \in H(w)$) iff there exists a $\Box$-lifting form $\triangle'$ of $\triangle$ such that $\triangle'\alpha \in Ext'_L(I)$ (resp. $\triangle' A \in Sat_L(I)$).*

This lemma can be proved by induction on the length of $\triangle$ in a straightforward way. We give below the main lemma of this subsection.

Lemma 5. *Let I be a model generator and $M = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ be the extended L-model graph of I. Then for any w and u such that $R_i(w, u)$ holds:*

- *if $\Box_i\alpha \in H(w)$ then $\alpha \in H(u)$,*
- *if $\alpha \in H(u)$ then $\Diamond_i\alpha \in H(w)$.*

Proof. Let $\{R'_j \mid 1 \leq j \leq m\}$ be the skeleton of M. We prove this lemma by induction on the number of steps needed to obtain $R_i(w, u)$ when extending $\{R'_j \mid 1 \leq j \leq m\}$ to $\{R_j \mid 1 \leq j \leq m\}$.

Consider the first assertion. Suppose that $\Box_i\alpha \in H(w)$. By Lemma 4, there exists a $\Box$-lifting form $\triangle$ of w such that $\triangle\Box_i\alpha \in Ext'_L(I)$. Since $R_i(w, u)$ holds, there are the following cases to consider:

- Case $u = w\langle E\rangle_i$ and $R'_i(w, w\langle E\rangle_i)$: The assertion holds by the definition of M.
- Case $AT(i)$ holds and $u = w$: Since $\triangle\Box_i\alpha \in Ext'_L(I)$, we have $\triangle\alpha \in Ext'_L(I)$, and by Lemma 4, $\alpha \in H(u)$.
- Case $AI(i,j)$ holds and $R_i(w,u)$ is created from $R_j(w,u)$: Since $\triangle\Box_i\alpha \in Ext'_L(I)$, we have $\triangle\Box_j\alpha \in Ext'_L(I)$, and by Lemma 4, $\Box_j\alpha \in H(w)$. Hence, by the inductive assumption, $\alpha \in H(u)$.
- Case $AB(j,i)$ holds and $R_i(w,u)$ is created from $R_j(u,w)$: Since $\Box_i\alpha \in H(w)$, by the inductive assumption, $\Diamond_j\Box_i\alpha \in H(u)$. By Lemma 4, there exists a $\Box$-lifting form $\triangle'$ of u such that $\triangle'\Diamond_j\Box_i\alpha \in Ext'_L(I)$. Thus $\triangle'\alpha \in Ext'_L(I)$. Hence, by Lemma 4, $\alpha \in H(u)$.
- Case $A4(i,j,k)$ holds and $R_i(w,u)$ is created from $R_j(w,v)$ and $R_k(v,u)$: Since $\triangle\Box_i\alpha \in Ext'_L(I)$, we have $\triangle\Box_j\Box_k\alpha \in Ext'_L(I)$, and by Lemma 4, $\Box_j\Box_k\alpha \in H(w)$. Hence, by the inductive assumption, $\Box_k\alpha \in H(v)$ and $\alpha \in H(u)$.
- Case $A5(j,k,i)$ holds and $R_i(w,u)$ is created from $R_j(v,u)$ and $R_k(v,w)$: Since $\Box_i\alpha \in H(w)$, by the inductive assumption, $\Diamond_k\Box_i\alpha \in H(v)$. Hence, by Lemma 4, there exists a $\Box$-lifting form $\triangle'$ of v such that $\triangle'\Diamond_k\Box_i\alpha \in Ext'_L(I)$. Hence $\triangle'\Box_j\alpha \in Ext'_L(I)$, and by Lemma 4, $\Box_j\alpha \in H(v)$. By the inductive assumption, it follows that $\alpha \in H(u)$.

The second assertion can be proved in a similar way (see [11]).

4.3 Remaining Proofs

We also need the following lemma (labeled Expected Lemma 2 in [13]), which states that the standard L-model of I is really an L-model of I.

Lemma 6. *Let I be a model generator, M the standard/extended L-model graph of I, and σ the standard $\Diamond$-realization function on M. Then M is an L-model and $M, \sigma \vDash I$.*

Proof. By Lemma 3, it suffices to prove for the case when M is the standard L-model graph of I. By the definition, M is an L-model. It can be proved by induction on the length of α that for any $w \in W$, if $\alpha \in H(w)$, then $M, \sigma, w \vDash \alpha$. The cases when α is a classical atom or $\alpha = \langle E\rangle_i\beta$ are trivial. The case when $\alpha = \Box_i\beta$ is solved by Lemmas 3 and 5. Hence $M, \sigma \vDash I$.

Proof of Lemma 1. Let I' be the least extension of I such that, if $\circledbox\varphi$ is a program clause of P, $\varphi = (A \leftarrow B_1, \ldots, B_n)$, and ψ is a ground instance of φ, then $\circledbox p_\psi \in I'$, where p_ψ is a fresh 0-ary predicate symbol. Let M and M' be the extended L-model graphs of I and I', respectively. It is easy to see that these model graphs have the same frame. Let $M = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ and $M' = \langle W, \tau, R_1, \ldots, R_m, H'\rangle$. Clearly, M is an L-model. By Lemma 3, it suffices to show that $M \vDash P$.

Let $\circledbox\varphi$ be a program clause of P, $\varphi = (A \leftarrow B_1, \ldots, B_n)$, and ψ a ground instance of φ. By Lemma 6, $M' \vDash \circledbox p_\psi$. To prove that $M \vDash P$ it is sufficient

to show that for any $w \in W$, if $p_\psi \in H'(w)$ then $M, w \vDash \psi$. Suppose that $p_\psi \in H'(w)$.

Let $\triangle = w$ and $\boxdot' = \Box_{i_1} \ldots \Box_{i_k}$ be a $\Box$-lifting form of $\triangle$. By Lemma 4, some $\Box$-lifting form of $\triangle p_\psi$ belongs to $Sat_L(I')$. This $\Box$-lifting form must be $\boxdot' p_\psi$. Thus $\boxdot' p_\psi \in Sat_L(\{\boxdot p_\psi\})$. Hence $\boxdot p_\psi \rightarrow \boxdot' p_\psi$ is L-valid and the program clause $\boxdot' \psi$ is a ground L-instance of $\boxdot \varphi$.

Let $\psi = (A' \leftarrow B'_1, \ldots, B'_n)$ and suppose that $M, w \vDash B'_i$ for all $1 \leq i \leq n$. We need to show that $M, w \vDash A'$. For this, we first show that a $\Box$-lifting form of $\triangle B'_i$ belongs to $Sat_L(I)$ for every $1 \leq i \leq n$. Consider the following cases:

- Case B'_i is a classical atom: The assertion follows from Lemma 4.
- Case B'_i is of the form $\Box_j E$: Since $M, w \vDash B'_i$, it follows that $M, w\langle\top\rangle_j \vDash E$, and by Lemma 4, some $\Box$-lifting form of $\triangle\langle\top\rangle_j E$ belongs to $Sat_L(I)$, which means that some $\Box$-lifting form of $\triangle B'_i$ belongs to $Sat_L(I)$.
- Case B'_i is of the form $\Diamond_j E$: Since $M, w \vDash B'_i$, there exists a world u such that $R_j(w, u)$ holds and $M, u \vDash E$. By Lemma 5, it follows that $\Diamond_j E \in H(w)$. Hence, by Lemma 4, some $\Box$-lifting form of $\triangle B'_i$ belongs to $Sat_L(I)$.

Therefore, by the definition of $T_{0L,P}$, some $\Box$-lifting form α of $\triangle A''$, where A'' is the forward labeled form of A', belongs to $T_{0L,P}(Sat_L(I))$. Since $T_{0L,P}(Sat_L(I)) = T_{L,P}(I) \subseteq I$, by Lemma 6, we have that $M, \sigma \vDash \alpha$, where σ is the standard $\Diamond$-realization function on M. Hence $M, w \vDash A'$. Thus $M, w \vDash \psi$, which completes the proof.

Proof of Lemma 2. Let $M' = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ be the extended L-model graph of I, $\boxdot = \Box_{i_1} \ldots \Box_{i_k}$ and $w = \langle\top\rangle_{i_1} \ldots \langle\top\rangle_{i_k}$. Suppose that α is of the form $\boxdot E$. Since $M \vDash \alpha$, by Lemma 3, we have $M', w \vDash E$. By Lemma 4, it follows that $\boxdot E \in Sat_L(I)$. Now suppose that α is of the form $\boxdot\Diamond_i E$. Since $M \vDash \alpha$, we have $M, w \vDash \Diamond_i E$, and by Lemma 3, $M', w \vDash \Diamond_i E$. There exists u such that $R_i(w, u)$ holds and $M', u \vDash E$. By Lemma 5, it follows that $\Diamond_i E \in H(w)$. Hence $\boxdot\Diamond_i E \in Sat_L(I)$ (by Lemma 4).

We have proved Lemmas 1 and 2, which completes the proof of Theorem 1.

5 Conclusions

We have developed semantics for MProlog programs in BSMM and proved that the given SLD-resolution calculus is sound and complete. The class BSMM of basic serial multimodal logics is much larger than the class of multimodal logics considered by Debart *et al.* using the translational approach [4] and is very different from the class of grammar modal logics considered by Baldoni *et al.* [2].

This paper is an extension of our previous paper on programming in monomodal logics [10] and is an instantiation of our general framework given in [13]. The SLD-resolution calculus for BSMM presented in this paper together with its soundness and completeness is, however, a strong and essential result.

The $Sat_L/rSat_L$ rules given for semantics of MProlog in BSMM are based directly on the axioms of the used modal logic. This makes our fixpoint semantics

and SLD-resolution calculus intuitive. The clarity of the rules suggests that our methods can be extended for other multimodal logics.

Our SLD-resolution calculus for MProlog in BSMM is elegant like a Hilbert-style axiom system, but similarly to using a Hilbert-style axiom system for automatic reasoning, it is not very efficient. For more specific modal logics, as reported in [12,13], we have implemented the modal logic programming system MProlog. It uses optimization techniques like normalization of modalities, better orderings of modal operators, options for restricting the search space.

In summary, we have successfully applied the direct approach for modal logic programming in a large class of basic multimodal logics, while not assuming any special restriction on the form of logic programs and goals.

References

1. Ph. Balbiani, L. Fariñas del Cerro, and A. Herzig. Declarative semantics for modal logic programs. In *Proceedings of the 1988 International Conference on Fifth Generation Computer Systems*, pages 507–514. ICOT, 1988.
2. M. Baldoni, L. Giordano, and A. Martelli. A framework for a modal logic programming. In *Joint International Conference and Symposium on Logic Programming*, pages 52–66. MIT Press, 1996.
3. M.J. Cresswell and G.E. Hughes. *A New Introduction to Modal Logic*. Routledge, 1996.
4. F. Debart, P. Enjalbert, and M. Lescot. Multimodal logic programming using equational and order-sorted logic. *Theoretical Comp. Science*, 105:141–166, 1992.
5. S. Demri. The complexity of regularity in grammar logics and related modal logics. *Journal of Logic and Computation*, 11(6):933–960, 2001.
6. L. Fariñas del Cerro. Molog: A system that extends Prolog with modal logic. *New Generation Computing*, 4:35–50, 1986.
7. M. Fitting and R.L. Mendelsohn. *First-Order Modal Logic*. Springer, 1998.
8. R.A. Kowalski. Predicate logic as a programming language. In J.L. Rosenfeld, editor, *Information Processing 74, Proc. of IFIP Congress 74*, pages 569–574, 1974.
9. J.W. Lloyd. *Foundations of Logic Programming, 2nd Ed.* Springer-Verlag, 1987.
10. L.A. Nguyen. A fixpoint semantics and an SLD-resolution calculus for modal logic programs. *Fundamenta Informaticae*, 55(1):63–100, 2003.
11. L.A. Nguyen. Multimodal logic programming and its applications to modal deductive databases. Manuscript (served as a technical report), available on Internet at `http://www.mimuw.edu.pl/~nguyen/papers.html`, 2003.
12. L.A. Nguyen. The modal logic programming system MProlog. In J.J. Alferes and J.A. Leite, editors, *Proceedings of JELIA 2004, LNCS 3229*, pages 266–278. Springer, 2004.
13. L.A. Nguyen. The modal logic programming system MProlog: Theory, design, and implementation. Available at `http://www.mimuw.edu.pl/~nguyen/mprolog`, 2005.
14. A. Nonnengart. How to use modalities and sorts in Prolog. In C. MacNish, D. Pearce, and L.M. Pereira, editors, *Proceedings of JELIA'94, LNCS 838*, pages 365–378. Springer, 1994.
15. H.J. Ohlbach. A resolution calculus for modal logics. In E.L. Lusk and R.A. Overbeek, editors, *Proc. of CADE-88, LNCS 310*, pages 500–516. Springer, 1988.
16. M.H. van Emden and R.A. Kowalski. The semantics of predicate logic as a programming language. *Journal of the ACM*, 23(4):733–742, 1976.