Reasoning about Epistemic States of Agents by Modal Logic Programming*

Linh Anh Nguyen

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

Abstract. Modal logic programming is one of appropriate approaches to deal with reasoning about epistemic states of agents. We specify here the least model semantics, the fixpoint semantics, and an SLD-resolution calculus for modal logic programs in the multimodal logic $KD4I_g5_a$, which is intended for reasoning about belief and common belief of agents. We prove that the presented SLD-resolution calculus is sound and complete. We also present a formalization of the wise men puzzle using a modal logic program in $KD4I_g5_a$. This shows that it is worth to study modal logic programming for multi-agent systems.

1 Introduction

Reasoning is an important aspect of agents. In order to be able to make right actions, an agent should have general knowledge of the field it works on, information about the environment, and abilities to interact with the environment, to make inferences, and to revise its knowledge base. In multi-agent systems, agents should be able to communicate, collaborate, and sometimes compete with each other. For this aim, an agent should have knowledge about other agents in the system and be able to reason about their epistemic states. It is not that an agent can have all information it wants or can reason exactly as the others, but at least it can simulate epistemic states of the other agents, using some assumptions. The wise men puzzle introduced by McCarthy [20] is an example of reasoning about epistemic states of agents. We will study it in Section 3.

Modal logics and logic programming are useful instruments for multi-agent systems. Using modal logics is a natural way to represent and reason about knowledge and belief of agents (see, e.g., [11, 33, 32, 17, 8, 1]). Logic programming is also useful because logical implication is probably the inference form humans use most and want to adopt for multi-agent systems. Thus, one can think about modal logic programming as an approach to deal with reasoning about epistemic states of agents.

* This is a revised version of: L.A. Nguyen. *Reasoning about Epistemic States of Agents by Modal Logic Programming.* In F. Toni and P. Torroni (eds), Proceedings of CLIMA VI, LNAI 3900, pages 37-56, Springer-Verlag, 2006.

Modal logic programming has been studied in a number of works (see the earlier surveys [29, 13] and the later works [28, 5, 22, 26]). There are two approaches: the direct approach [12, 3, 5, 22, 26] and the translation approach [9, 28]. The first approach directly uses modalities, while the second one translates modal logic programs to classical logic programs. In this paper we will use the direct approach. This approach is justifiable, as the direct approach deals with modalities more closely, and modalities allow us to separate object-level and epistemic-level notions nicely.

In [22], 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. In that work we do not assume any special restriction on occurrences of $\Box$ and $\Diamond$ in programs and goals. In [26], we generalized the methods of [22] and gave a general framework for developing fixpoint semantics, the least model semantics, and SLD-resolution calculi for logic programs in normal multimodal logics whose frame restrictions consist of the conditions of seriality and some classical first-order Horn formulas.

In this work, we instantiate the above mentioned framework for the multimodal logic $KD4I_g5_a$, which was introduced in [23] for reasoning about belief and common belief. We prove that the obtained SLD-resolution calculus is sound and complete. We also give a purely logical formalization of the wise men puzzle using a modal logic program in $KD4I_g5_a$.

The rest of this paper is structured as follows. In Section 2, we give definitions for multimodal logics, define the multimodal logic $KD4I_g5_a$ and the modal logic programming language MProlog. In Section 3, we recall the wise men puzzle and formalize it by an MProlog program in $KD4I_g5_a$. In Section 4, we instantiate the framework given in [26] for $KD4I_g5_a$ in order to specify the least model semantics, the fixpoint semantics, and an SLD-resolution calculus for MProlog programs in $KD4I_g5_a$. Soundness and completeness of the obtained SLD-resolution calculus is proved in Section 5. In Section 6, we give two more examples illustrating the usefulness of modal logic programming for multi-agent systems. In Section 7, we briefly mention related works and discuss how to extend our system to deal with actions and time. Finally, Section 8 contains some concluding remarks.

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 an emphasis 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 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 the world u is accessible from the world w via R_i.

A *fixed-domain Kripke model with rigid terms*, hereafter simply called a (Kripke) 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 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 $p^{M,w}$. (We adopt here the version with fixed-domain and rigid terms, as it is most popular. This work can be extended for other versions of Kripke semantics, e.g. with varying domain and flexible terms; see a discussion in [26].)

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* $M, V, w \models \psi$ for a formula ψ is defined as follows:

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

and as usual for 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 $M \models \varphi$, if $M, V, \tau \models \varphi$ for every V. For a set Γ of formulas, we call M a model of Γ and write $M \models \Gamma$ if $M \models \varphi$ for every $\varphi \in \Gamma$.

If as the class of admissible interpretations we take the class of all Kripke models (with no restrictions on the accessibility relations) then we obtain 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 $\forall x \, \exists y \, R_i(x, y)$.

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 $\Gamma \models_L \varphi$ and call φ a *logical consequence* of Γ in L if φ is true in every L-model of Γ.

2.2 The Multimodal Logic $KD4I_g5_a$

Suppose that there are n agents and $m = 2^n - 1$. Let g be an one-to-one function that maps every natural number less than or equal to m to a non-empty subset of $\{1, \ldots, n\}$. Suppose that an index $1 \leq i \leq m$ stands for the group of agents whose indices form the set $g(i)$. To capture belief and common belief of agents, we can extend K_m with the following axioms

- $(D) : \Box_i\varphi \rightarrow \neg\Box_i\neg\varphi$ (belief is consistent),
- $(4) : \Box_i\varphi \rightarrow \Box_i\Box_i\varphi$ (belief satisfies positive introspection),
- $(I_g) : \Box_i\varphi \rightarrow \Box_j\varphi$ if $g(i) \supset g(j)$ (if i indicates a supergroup of a group j then every common belief of i is also a common belief of j).
- $(5_a) : \neg\Box_i\varphi \rightarrow \Box_i\neg\Box_i\varphi$ if $g(i)$ is a singleton (belief of a single agent satisfies negative introspection).

Thus, for reasoning about belief and common belief, we can use:

$$KD4I_g5_a = K_m + (D) + (4) + (I_g) + (5_a)$$

Here we want to catch the most important properties of belief and common belief, and the aim is not to give an exact formulation of belief or common belief. The logic $KD4I_g5_a$ was introduced in [23]. It is different in the nature from the well-known multimodal logic of common knowledge. It also differs from the modal logic with mutual belief [1].

In [15] (an extension of [14]), Goré and Nguyen show that the satisfiability problem in the propositional version of $KD4I_g5_a$ is in EXPTIME. Clearly, the problem is PSPACE-hard (as $KD4I_g5_a$ contains $KD4$). We guess that the problem is EXPTIME-hard when $n \geq 3$ (i.e. $m \geq 7$). It is an open problem.

The given axioms correspond to the following frame restrictions:

Axiom	Corresponding Condition
(D)	$\forall u\, \exists v\, R_i(u, v)$
(4)	$\forall u, v, w\, (R_i(u, v) \wedge R_i(v, w) \rightarrow R_i(u, w))$
(I_g)	$R_j \subseteq R_i$ if $g(i) \supset g(j)$
(5_a)	$\forall u, v, w\, (R_i(u, v) \wedge R_i(u, w) \rightarrow R_i(w, v))$ if $g(i)$ is a singleton

For further reading on epistemic logics, see, e.g., [11, 33, 8, 1].

2.3 Modal Logic Programs

A *modality* is a (possibly empty) sequence of modal operators. A *universal modality* is a modality which contains 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 a 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 and A, B_1, ..., B_n to denote formulas of the form E, $\Box_i E$, or $\Diamond_i E$.

A *program clause* is a formula of the form $\boxdot(A \leftarrow B_1, \ldots, B_n)$, where $n \geq 0$. $\boxdot$ 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. The *empty goal* (i.e. the *empty clause*) is denoted by $\diamond$.

In $KD4I_g5_a$, if $g(i)$ is a singleton then we have the equivalence $\nabla_i\nabla'_i\varphi \equiv \nabla'_i\varphi$ for any modal operators ∇_i and ∇'_i with the same modal index i. For this reason, we adopt some restrictions to simplify the form of MProlog programs and goals in $KD4I_g5_a$. An MProlog program is called a *$KD4I_g5_a$-MProlog program* if the modal contexts of its program clauses do not contain subsequences of the form $\Box_i\Box_i$ if $g(i)$ is a singleton. An MProlog goal is called a *$KD4I_g5_a$-MProlog goal* if each of its goal atoms $\triangle E$ satisfies the condition that $\triangle$ does not contain subsequences of the form $\Box_i\Box_i$ or $\Box_i\Diamond_i$ if $g(i)$ is a singleton.

Let P be an $KD4I_g5_a$-MProlog program and $G = \leftarrow \alpha_1, \ldots, \alpha_k$ be an $KD4I_g5_a$-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 $KD4I_g5_a$ for $P \cup \{G\}$ if θ is an answer for $P \cup \{G\}$ and $P \models_{KD4I_g5_a} \forall((\alpha_1 \wedge \ldots \wedge \alpha_k)\theta)$.

It is shown in [23] that MProlog has the same expressiveness power as the general Horn fragment in normal modal logics. Moreover, the restrictions adopted for $KD4I_g5_a$-MProlog do not reduce expressiveness of the language (see [23]).

3 The Wise Men Puzzle

Before considering technical details of semantics of $KD4I_g5_a$-MProlog, we give a formalization of the three wise men puzzle in MProlog. The puzzle is a famous benchmark introduced by McCarthy [20] for AI. It can be stated as follows (cf. [18]). A king wishes to know whether his three advisors (A, B, C) are as wise as they claim to be. Three chairs are lined up, all facing the same direction, with one behind the other. The wise men are instructed to sit down in the order A, B, C. Each of the men can see the backs of the men sitting before them (e.g. C can see A and B). The king informs the wise men that he has three cards, all of which are either black or white, at least one of which is white. He places one card, face up, behind each of the three wise men, explaining that each wise man must determine the color of his own card. Each wise man must announce the color of his own card as soon as he knows what it is. All know that this will happen. The room is silent; then, after a while, wise man A says "My card is white!".

The wise men puzzle has been previously studied in a number of works (e.g., [20, 18, 10, 7, 2, 28, 4]). McCarthy [20] directly used possible worlds to formalize the puzzle. Konolige [18], Nonnengart [28], and Baldoni [4] also used modal logics for the puzzle. Konolige [18] focused on limited reasoning, Nonnengart [28] used semi-functional translation for modal logic programming, and Baldoni [4] used a prefixed tableau system. Both McCarthy [20] and Nonnengart [28] used some feature of mutual belief, but they did not define it purely. Baldoni [4] adopted too strong versions of axioms 4 and 5, which are rather not suitable for the puzzle. As other approaches for the wise men puzzle, Elgot-Drapkin [10] used step-logics, while Cimatti and Serafini [7], Attardi and Simi [2] studied reasoning in belief-contexts. Our formalization of the wise men puzzle given below uses $KD4I_g5_a$-MProlog. It is more elegant than the above-mentioned formalizations, as it uses a modal logic with a clear semantics of common belief in a direct way.

As reported in [24], we have designed and implemented a modal logic programming system, also called MProlog. In that system, SLD-resolution calculi for MProlog can be specified according to the theoretical framework given in [26]. An instantiation of that framework for $KD4I_g5_a$ is presented in the next section. Its implementation (of SLD-resolution) is denoted by ccKD4Ig5a. In that implementation, *bel* denotes belief and *pos* denotes possibility, and modalities are represented by lists, e.g. $\Box_i\langle X\rangle_j\Diamond_k q(a)$ is represented by $[bel(I), pos(J, X), pos(K)] : q(a)$. The implemented calculus requires definitions of predicates $singleton\_group/1$, $subgroup/2$, and $union\_group/3$. Denote the wise men by a, b, c, and the possible groups by gAB, gAC, gBC, $gABC$, where, e.g., $gABC = \{a, b, c\}$. Thus, $[bel(gABC)] : \varphi$ means that φ is a common belief of the group $\{a, b, c\}$. Define the mentioned required predicates in the usual way. The three wise men problem can be formalized by the following program:

```
:- calculus ccKD4Ig5a.

% If Y sits behinds X then X's card is white if Y considers this as possible.
[bel(gABC)]: (white(X) :-
        member(X, [a,b,c]), member(Y, [a,b,c]), X @< Y, [pos(Y)]:white(X)).

% The following formula is "dual" to the above formula.
[bel(gABC)]: ([bel(Y)]:black(X) :-
        member(X, [a,b,c]), member(Y, [a,b,c]), X @< Y, black(X)).

% At least one of the wise men has a white card.
[bel(gABC)]: (white(a) :- black(b), black(c)).
[bel(gABC)]: (white(b) :- black(c), black(a)).
[bel(gABC)]: (white(c) :- black(a), black(b)).

/* Each of B and C does not know the color of his own card. In particular, each
of the men considers that it is possible that his own card is black. */
[bel(gABC),pos(b)]:black(b).
[bel(gABC),pos(c)]:black(c).
```

The question is whether A believes that his card is white. It is passed to the interpreter as $mcall([bel(a)] : white(a))$ and solved in less than 1 second[1] using certain option settings.

The above program uses the syntax of the implemented system. We give below a version using the purely logical formalism of MProlog. For clarity, instead of numeric indices we use a, b, c, ab, ac, bc, abc with the meaning that $g(a) = \{a\}$, $g(b) = \{b\}$, $g(c) = \{c\}$, ..., and $g(abc) = \{a, b, c\}$. Let P_{wise_men} be the following program:

$$\begin{aligned}
\varphi_1 &= \Box_{abc}\,(white(a) \leftarrow \Diamond_b\, white(a))\\
\varphi_2 &= \Box_{abc}\,(white(a) \leftarrow \Diamond_c\, white(a))\\
\varphi_3 &= \Box_{abc}\,(white(b) \leftarrow \Diamond_c\, white(b))\\
\varphi_4 &= \Box_{abc}\,(\Box_b\, black(a) \leftarrow black(a))\\
\varphi_5 &= \Box_{abc}\,(\Box_c\, black(a) \leftarrow black(a))\\
\varphi_6 &= \Box_{abc}\,(\Box_c\, black(b) \leftarrow black(b))\\
\varphi_7 &= \Box_{abc}\,(white(a) \leftarrow black(b), black(c))\\
\varphi_8 &= \Box_{abc}\,(white(b) \leftarrow black(c), black(a))\\
\varphi_9 &= \Box_{abc}\,(white(c) \leftarrow black(a), black(b))\\
\varphi_{10} &= \Box_{abc}\Diamond_b\, black(b)\\
\varphi_{11} &= \Box_{abc}\Diamond_c\, black(c)
\end{aligned}$$

The goal is $\leftarrow \Box_a white(a)$. We will continue this example in Section 4.5.

For a formalization of the puzzle with n wise men, see [26].

4 Semantics of $KD4I_g5_a$-MProlog Programs

In this section, we present the least model semantics, the fixpoint semantics and an SLD-resolution calculus for $KD4I_g5_a$-MProlog programs. For abbreviation, *from now on we use L to denote* $KD4I_g5_a$.

4.1 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 $\triangle\Diamond_i E$, 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[2];
- E, F : classical atoms (which may contain variables) or $\top$;

[1] on TravelMate 230X, 1.7GHz-M

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

– 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 $EdgeLabels = \{\langle E\rangle_i \mid E \in \mathcal{B} \cup \{\top\} \text{ and } 1 \leq i \leq m\}$, where $\mathcal{B}$ is the Herbrand base (i.e. the set of all ground classical atoms). The semantics of $\langle E\rangle_i \in EdgeLabels$ 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 $\sigma : W \times EdgeLabels \to W$ such that if $\sigma(w, \langle E\rangle_i) = u$, then $R_i(w, u)$ holds and $M, u \models E$. Given a $\Diamond$-realization function σ, a world $w \in W$, and a ground formula φ, the satisfaction relation $M, \sigma, w \models \varphi$ is defined in the usual way, except that $M, \sigma, w \models \langle E\rangle_i\psi$ iff $\sigma(w, \langle E\rangle_i)$ is defined and $M, \sigma, \sigma(w, \langle E\rangle_i) \models \psi$. We write $M, \sigma \models \varphi$ to denote that $M, \sigma, \tau \models \varphi$. For a set I of ground atoms, we write $M, \sigma \models I$ to denote that $M, \sigma \models \alpha$ for all $\alpha \in I$; we write $M \models I$ and call M a model of I if $M, \sigma \models I$ for *some* σ.

4.2 Model Generators

We define that a modality $\nabla^{(1)}_{i_1} \ldots \nabla^{(k)}_{i_k}$ is in the *L-normal form* if for all $1 \leq j < k$ if $g(i_j)$ is a singleton then $i_j \neq i_{j+1}$. (Note that if $g(i)$ is a singleton then $\nabla_i\nabla'_i\varphi \equiv \nabla'_i\varphi$ is $KD4I_g5_a$-valid.) A modality is in *L-normal labeled form* if it is in L-normal form and does not contain modal operators of the form $\Diamond_i$ or $\langle\top\rangle_i$. An atom is in *L-normal (labeled) form* if it is of the form $\triangle E$ with $\triangle$ in L-normal (labeled) form. An atom is in *almost L-normal labeled form* if it is of the form $\triangle A$ with $\triangle$ in L-normal labeled form.

A *model generator* is a set of ground atoms not containing $\Diamond_i$, $\langle\top\rangle_i$, $\top$. An *L-normal model generator* is a model generator consisting of atoms in L-normal labeled form.

We will define the *standard L-model* of an L-normal 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 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.

$L = KD4I_g5_a$, $\quad$ L-MProlog	
$\preceq_L$ is defined by Definition **??** at page **??**. A modality $\nabla_{i_1}^{(1)} \dots \nabla_{i_k}^{(k)}$ is in *L-normal form* if $\quad$ for all $1 \le j < k$ if $g(i_j)$ is a singleton then $i_j \neq i_{j+1}$.	
Rules specifying operators Ext_L, Sat_L, NF_L, rNF_L, $rSat_L$:	
Ext_L	$\triangle\Box_i\alpha \to \triangle\Box_j\alpha$ if $g(i) \supset g(j)$ (1) $\triangle\Box_i\alpha \to \triangle\Box_i\Box_i\alpha$ (2) $\triangle\nabla_i\Box_i\alpha \to \triangle\Box_i\alpha$ if $g(i)$ is a singleton (3)
Sat_L	the rules specifying Ext_L plus $\triangle\langle F\rangle_i E \to \triangle\Box_i\Diamond_i E$ if $g(i)$ is a singleton (4) $\triangle\nabla\nabla' E \to \triangle\Diamond_i E$ if $\Diamond_i \preceq_L \nabla$ and $\Diamond_i \preceq_L \nabla'$ (5)
NF_L	$\triangle\nabla_i\nabla'_i E \to \triangle\nabla'_i E$ if $g(i)$ is a singleton and $\quad\nabla'_i$ is of the form $\Box_i$ or $\langle E\rangle_i$ (6)
rNF_L	$\triangle\nabla_i E \leftarrow \triangle\langle X\rangle_i\nabla_i E$ if $g(i)$ is a singleton, $\quad\nabla_i$ is of the form $\Box_i$ or $\langle E\rangle_i$, and X is a fresh atom variable (7)
$rSat_L$	$\triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_i E$ for X being a fresh atom variable (8) $\triangle\Diamond_i E \leftarrow \triangle\Diamond_j E$ if $g(i) \supset g(j)$ (9) $\triangle\nabla_i\alpha \leftarrow \triangle\Box_j\alpha$ if $g(i) \subseteq g(j)$ (10) $\triangle\Box_i\Box_i\alpha \leftarrow \triangle\Box_i\alpha$ (11) $\triangle\Box_i\alpha \leftarrow \triangle\langle X\rangle_i\Box_i\alpha$ if $g(i)$ is a singleton and $\quad X$ is a fresh atom variable (12) $\triangle\nabla_i\Diamond_i E \leftarrow \triangle\Diamond_i E$ if $g(i)$ is a singleton (13) $\triangle\Diamond_i E \leftarrow \triangle\langle X\rangle_j\Diamond_i E$ if $g(i) \supseteq g(j)$ and $\quad X$ is a fresh atom variable (14)

Table 1. A schema for semantics of $KD4I_g5_a$-MProlog

A *forward rule* is a schema of the form $\alpha \to \beta$, 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 1. Given an L-normal model generator I, $Ext_L(I)$ is the least extension of I that contains all ground atoms in L-normal labeled form that are derivable from some atom of I using the rules specifying Ext_L. Note that $Ext_L(I)$ is an L-normal model generator if so is I.

Denote $Serial_L = \{\boxdot\langle\top\rangle_i\top \mid 1 \le i \le m$ and $\boxdot\langle\top\rangle_i$ is in L-normal form$\}$.

Let I be an L-normal model generator. The *standard L-model* of I is constructed by building an L-model for $Ext_L(I) \cup Serial_L$ according to the semantics of ground labeled modal operators, and formally 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 $1 \leq i \leq m\})$, $\tau = \epsilon$, $H(\tau) = Ext_L(I) \cup Serial_L$. 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[3] 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 $Serial_L$). 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 can be shown that *the standard L-model of an L-normal model generator I is a least L-model of I.*

4.3 Fixpoint Semantics

We now consider the direct consequence operator $T_{L,P}$. Given an L-normal model generator I, how can $T_{L,P}(I)$ be defined? Based 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 $T_{_0L,P}$. The set $T_{_0L,P}(Sat_L(I))$ is a model generator but not necessary in L-normal form. Finally, the *normalization operator* NF_L converts $T_{_0L,P}(Sat_L(I))$ to an L-normal model generator. $T_{L,P}(I)$ is defined as $NF_L(T_{_0L,P}(Sat_L(I)))$.

To compare modal operators we define $\preceq_L$ to be the least reflexive and transitive relation between modal operators such that $\Diamond_i \preceq_L \langle E\rangle_i \preceq_L \Box_i$, $\Diamond_i \preceq_L \langle X\rangle_i \preceq_L \Box_i$, and if $g(i) \subseteq g(j)$ then $\Box_i \preceq_L \Box_j$ and $\Diamond_j \preceq_L \Diamond_i$.

An atom $\nabla^{(1)} \ldots \nabla^{(n)} \alpha$ is called an *L-instance* of an atom $\nabla^{(1')} \ldots \nabla^{(n')} \alpha'$ if there exists a substitution θ such that $\alpha = \alpha'\theta$ and $\nabla^{(i)} \preceq_L \nabla^{(i')}\theta$ for all $1 \leq i \leq n$ (treating $\nabla^{(i')}$ as an expression). For example, if $g(1) \subseteq g(2)$ then $\Box_1\Diamond_2 E$ is an L-instance of $\Box_2\langle F\rangle_1 E$.

A modality $\triangle$ is called an *L-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 L-instance of $\triangle' E$ for some ground classical atom E.

Let $\boxdot$ and $\boxdot'$ be universal modalities in L-normal form. We say that $\boxdot$ is an *L-context instance* of $\boxdot'$ if $\boxdot'\varphi \rightarrow \boxdot\varphi$ is L-valid (for every φ). (It can be shown that the propositional version of the logic L is decidable. So, the problem of checking whether a given universal modality is an L-context instance of another one is also decidable.)

Let $\boxdot$ and $\boxdot'$ be universal modalities in L-normal form, φ and φ' be program clauses with empty modal context. We say that $\boxdot\varphi$ is an *L-instance* of (a program

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

clause) $\boxdot'\varphi'$ if $\boxdot$ is an L-context instance of $\boxdot'$ and there exists a substitution θ such that $\varphi = \varphi'\theta$.

For example, if $g(1) \subseteq g(2)$ then $\Box_2\Box_1$ is an L-context instance of $\Box_2$ and $\Box_2\Box_1(p(a) \leftarrow q(a))$ is an L-instance of $\Box_2(p(x) \leftarrow q(x))$.

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

The *saturation operator* Sat_L is specified by the corresponding forward rules given in Table 1. Given an L-normal model generator I, $Sat_L(I)$ is the least extension of I that contains all ground atoms in almost L-normal labeled form that are derivable from some atom in I using the rules specifying Sat_L. For example, if $g(1)$ is a singleton and $g(2)$ is not, then $\Box_2\Box_2\Box_1\Diamond_1 p(a) \in Sat_L(\{\Box_2\langle q(b)\rangle_1\, p(a)\})$.

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 $\alpha' = \alpha$. For example, the forward labeled form of $\Diamond_1 s(a)$ is $\langle s(a)\rangle_1 s(a)$.

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 L-normal 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[4] ground modality in L-normal labeled form such that $\triangle$ is an L-instance of $\boxdot$ and $\triangle B_i$ is an L-instance of some atom of I (for every $1 \leq i \leq n$), then the forward labeled form of $\triangle A$ belongs to $T_{0L,P}(I)$.

For example, if $g(1) \subseteq g(2)$ and P contains the clause $\Box_2(\Diamond_1 p(x) \leftarrow q(x), r(x),\ \Box_1 s(x),\ \Diamond_2 t(x))$ and $I = \{\langle q(a)\rangle_1 q(a),\ \langle q(a)\rangle_1 r(a),\ \Box_2\Box_2 s(a),\ \Box_2\langle t(a)\rangle_1 t(a)\}$, then $\langle q(a)\rangle_1\langle p(a)\rangle_1 p(a) \in T_{0L,P}(I)$.

The *normalization operator* NF_L is specified by the corresponding forward rules given in Table 1. Given a model generator I, $NF_L(I)$ is the set of all ground atoms in L-normal labeled form that are derivable from some atom of I using the rules specifying NF_L. For example, if $g(1)$ is a singleton then $NF_L(\{\langle q(a)\rangle_1\langle p(a)\rangle_1 p(a)\}) = \{\langle p(a)\rangle_1 p(a)\}$.

Define $T_{L,P}(I) = NF_L(T_{0L,P}(Sat_L(I)))$. By definition, the operators Sat_L, $T_{0L,P}$, and NF_L are all 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 can be shown that for an L-MProlog program P, $M_{L,P}$ *is a least* L*-model of* P. See also Lemma 1 given in Section 5.

4.4 SLD-Resolution

The main work in developing an SLD-resolution calculus for L-MProlog is to specify a reverse analogue of the operator $T_{L,P}$. The operator $T_{L,P}$ is a com-

[4] w.r.t. the pre-order between modalities described earlier for L

position of Sat_L, $T_{_0L,P}$, and NF_L. So, we have to investigate reversion of these operators.

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 L-normal labeled form, is called the *selected atom*, and A' is called the *selected head atom*;
- $\triangle'$ is an L-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 $\leftarrow (\alpha_1, \ldots, \alpha_{i-1}, \triangle' B_1, \ldots, \triangle' B_n, \alpha_{i+1}, \ldots, \alpha_k)\theta$.

For example, if $g(1) \subseteq g(2)$ then $\leftarrow \Box_1 \Diamond_2 q(x), \Box_1 r(x)$ is an L-resolvent of $\leftarrow \Box_1 p(x)$ and $\Box_2(p(x) \leftarrow \Diamond_2 q(x), r(x))$ (here, $\boxdot = \Box_2$ and $\triangle' = \boxdot' = \Box_1$).

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 1. 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".

As a reverse analogue of the operator NF_L, we provide the operator rNF_L, which is specified by the corresponding backward rules given in Table 1. We say that $\beta =_\theta rNF_L(\alpha)$ *using an* rNF_L *rule* $\alpha' \leftarrow \beta'$ if θ is an mgu such that $\alpha\theta \leftarrow \beta$ is of the form $\alpha' \leftarrow \beta'$. We write $\beta =_\theta rNF_L(\alpha)$ to denote that "$\beta =_\theta rNF_L(\alpha)$ using some rNF_L rule". For example, if $g(1)$ is a singleton then we have $\langle Y\rangle_1\langle E\rangle_1 E =_\theta rNF_L(\langle X\rangle_1 E)$ with $\theta = \{X/E\}$ and Y being a fresh atom variable.

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.

Similarly, G' is *derived* from G and an rNF_L rule φ using an mgu θ, and called an *(L-)resolvent* of G and φ, if α_i is called the *selected atom*, $\alpha'_i =_\theta rNF_L(\alpha_i)$ using φ, and $G' = \leftarrow \alpha_1\theta, \ldots, \alpha_{i-1}\theta, \alpha'_i, \alpha_{i+1}\theta, \ldots, \alpha_k\theta$.

For example, resolving $\leftarrow \Box_1\Box_1 p(x)$ with the rule $\triangle\Box_i\Box_i\alpha \leftarrow \triangle\Box_i\alpha$ results in $\leftarrow \Box_1 p(x)$, since $\triangle$ is instantiated to the empty modality, i is instantiated to 1, and α is instantiated to $p(x)$.

Observe that $rSat_L$ rules and rNF_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 for $rSat_L$ rules.

We now define SLD-derivation and SLD-refutation.

Let P be an L-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, $rSat_L$ rules, or rNF_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 or an rNF_L rule 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 [19]).

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

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

4.5 Example

We give here an SLD-refutation of $P_{wise\_men} \cup \{\leftarrow \Box_a white(a)\}$ in $KD4I_g5_a$, where P_{wise_men} is the $KD4I_g5_a$-MProlog program given in Section 3.

Goals	Input clauses/rules	MGUs
$\leftarrow \Box_a\, white(a)$		
$\leftarrow \Box_a \Diamond_b\, white(a)$	φ_1	
$\leftarrow \Box_a \langle X_2 \rangle_b\, white(a)$	(8)	
$\leftarrow \Box_a \langle X_2 \rangle_b \Diamond_c\, white(a)$	φ_2	
$\leftarrow \Box_a \langle X_2 \rangle_b \langle X_4 \rangle_c\, white(a)$	(8)	
$\leftarrow \Box_a \langle X_2 \rangle_b \langle X_4 \rangle_c\, black(b), \Box_a \langle X_2 \rangle_b \langle X_4 \rangle_c\, black(c)$	φ_7	
$\leftarrow \Box_a \langle X_2 \rangle_b\, black(b), \Box_a \langle X_2 \rangle_b \langle X_4 \rangle_c\, black(c)$	φ_6	
$\leftarrow \Box_a \langle black(b) \rangle_b \langle X_4 \rangle_c\, black(c)$	φ_{10}	$\{X_2/black(b)\}$
$\diamond$	φ_{11}	$\{X_4/black(c)\}$

5 Soundness and Completeness

In this section, we prove soundness and completeness of the SLD-resolution calculus given for $KD4I_g5_a$-MProlog, which is stated as follows.

Theorem 1. *Let P be an $KD4I_g5_a$-MProlog program and G an $KD4I_g5_a$-MProlog goal. Then every computed answer in $KD4I_g5_a$ of $P \cup \{G\}$ is a correct answer in $KD4I_g5_a$ of $P \cup \{G\}$. Conversely, for every correct answer θ in $KD4I_g5_a$ of $P \cup \{G\}$, there exists a computed answer γ in $KD4I_g5_a$ of $P \cup \{G\}$ such that $G\theta = G\gamma\delta$ for some substitution δ.*

In [26], 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 $KD4I_g5_a$-MProlog presented in the previous section and summarized in Table 1 is based on and compatible with the framework given in [26].

By the results of [26], to prove soundness and completeness of SLD-resolution of $KD4I_g5_a$-MProlog, we can prove Expected Lemmas 4 – 10 of [26] (w.r.t. the

schema given in Table 1). The Expected Lemma 6 is trivial, and the Expected Lemmas 7 – 10, which concern properties of the operators Sat_L, NF_L, $rSat_L$, and rNF_L, can be verified in a straightforward way. The remaining Expected Lemmas 4 and 5 are renumbered respectively as Lemmas 1 and 2 given below.

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

To prove these lemmas we need Lemmas 3 and 4 given below.

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 3. *Let I be an L-normal model generator and $M = \langle W, \tau, R_1, \ldots, R_m, H\rangle$ the standard 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 not containing $\top$, $\alpha \in H(w)$ iff there exists a $\Box$-lifting form $\triangle'$ of $\triangle$ such that $\triangle'\alpha \in Ext_L(I)$.*

This lemma can be easily proved by induction on the length of $\triangle$.

The following lemma is labeled Expected Lemma 2 in [26]. It states that the standard L-model of I is really an L-model of I.

Lemma 4. *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 $M, \sigma \models I$.*

Proof. By the definition, M is an L-model. Let $\{R'_i \mid 1 \leq i \leq m\}$ 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 $M, \sigma, w \models \alpha$. The cases when α is a classical atom or $\alpha = \langle E\rangle_i\beta$ are trivial. Consider the remaining case when $\alpha = \Box_i\beta$. Let u be a world such that $R_i(w, u)$ holds. We show that $\beta \in H(u)$ by induction on the derivation of $R_i(w, u)$:

- The case $R'_i(w, u)$ holds is trivial.
- Case $R_i(w, u)$ is derived from $R_j(w, u)$ with $j < i$: Treating w as a modality, by Lemma 3, a $\Box$-lifting form of $w\Box_i\beta$ belongs to $Ext_L(I)$. By the definition of Ext_L, a $\Box$-lifting form of $w\Box_j\beta$ belongs to $Ext_L(I)$. By Lemma 3, it follows that $\Box_j\beta \in H(w)$. Hence, by the inductive assumption, $\beta \in H(u)$.
- Case $R_i(w, u)$ is derived from $R_j(w, v)$ and $R_k(v, u)$ with $j \leq i$ and $k \leq i$: By Lemma 3, a $\Box$-lifting form of $w\Box_i\beta$ belongs to $Ext_L(I)$. By the definition of Ext_L, a $\Box$-lifting form of $w\Box_j\Box_k\beta$ belongs to $Ext_L(I)$. By Lemma 3, it follows that $\Box_j\Box_k\beta \in H(w)$. Hence, by the inductive assumption, $\Box_k\beta \in H(v)$ and $\beta \in H(u)$.

– Case $g(i)$ is a singleton and $R_i(w,u)$ is derived from $R'_i(v,w)$ and $R'_k(v,u)$: By Lemma 3, a $\Box$-lifting form of some $v\nabla_i\Box_i\beta$ belongs to $Ext_L(I)$. By the definition of Ext_L, a $\Box$-lifting form of $v\Box_i\beta$ belongs to $Ext_L(I)$. By Lemma 3, it follows that $\Box_i\beta \in H(v)$. Hence, by the inductive assumption, $\beta \in H(u)$.

Proof of Lemma 1 Let M be the standard L-model of I and σ the standard $\Diamond$-realization function on M. By the definition of L-instances of program clauses and the construction of M, it is sufficient to prove that for any ground L-instance $\boxdot(A \leftarrow B_1, \ldots, B_n)$ of some program clause of P, for any $w \in W$ being an L-instance of $\boxdot$, $M, w \models (A \leftarrow B_1, \ldots, B_n)$. Suppose that $M, w \models B_i$ for all $1 \leq i \leq n$. We show that $M, w \models A$.

Let $\triangle = w = \langle E_1\rangle_{i_1} \ldots \langle E_k\rangle_{i_k}$. We first show that for any ground simple atom B of the form E, $\Box_i E$, or $\Diamond_i E$, if $M, w \models B$ then $\triangle B$ is an L-instance of some atom from $Sat_L(I)$. Suppose that $M, w \models B$. If $k \geq 1$ and $i = i_k$ and $g(i)$ is a singleton, then let $v = \langle E_1\rangle_{i_1} \ldots \langle E_{k-1}\rangle_{i_{k-1}}$, else let $v = w$.

If $B = E$, then by Lemma 3, some $\Box$-lifting form of $\triangle B$ belongs to $Ext_L(I)$, and hence $\triangle B$ is an L-instance of some atom from $Sat_L(I)$.

Now suppose that $B = \Box_i E$. Let $u = v\langle\top\rangle_i$ and $\triangle' = v\Box_i$. We have $R_i(w,u)$, and hence $M, u \models E$. By Lemma 3, it follows that some $\Box$-lifting form of $\triangle' E$ belongs to $Ext_L(I)$. Hence, $\triangle B$ is an L-instance of some atom from $Sat_L(I)$.

Next, suppose that $B = \Diamond_i E$. Consider the case $w \neq v$ (i.e. $i = i_k$ and $g(i)$ is a singleton). Since $M, w \models B$, there exists F such that $v\langle F\rangle_i$ is a world of M and $M, v\langle F\rangle_i \models E$. Let $\triangle' = v\langle F\rangle_i$. By Lemma 3, some $\Box$-lifting form of $\triangle' E$ belongs to $Ext_L(I)$. Hence, by the rules (2) and (4) of Sat_L, $\triangle B$ is an L-instance of some atom from $Sat_L(I)$. Now consider the case $w = v$ (i.e. $k = 0$ or $i \neq i_k$ or $g(i)$ is not a singleton). Since $M, w \models \Diamond_i E$, as shown in the proof of Expected Lemma 2, there exists $u = w\langle F_1\rangle_{j_1} \ldots \langle F_h\rangle_{j_h}$ such that $M, u \models E$, $h \geq 1$, and $g(l) \subseteq g(i)$ for all $l \in \{j_1, \ldots, j_h\}$. By Lemma 3, some $\Box$-lifting form of $w\langle F_1\rangle_{j_1} \ldots \langle F_h\rangle_{j_h} E$ belongs to $Ext_L(I)$. It follows that some $\Box$-lifting form of $\triangle\langle F_1\rangle_{j_1} \ldots \langle F_h\rangle_{j_h} E$ belongs to $Sat_L(I)$. By the rules of Sat_L, some $\Box$-lifting form of $\triangle\Diamond_i E$ belongs to $Sat_L(I)$. Hence $\triangle B$ is an L-instance of some atom from $Sat_L(I)$.

Since $M, w \models B_i$ for $1 \leq i \leq n$, it follows that $\triangle B_i$ is an L-instance of some atom from $Sat_L(I)$. Consequently, $\triangle A$ is an L-instance of some atom α from $T_{0L,P}(Sat_L(I))$. Let α' be the L-normal form of α, i.e. $NF_L(\{\alpha\}) = \{\alpha'\}$. We have $\alpha' \in T_{L,P}(I) \subseteq I$. By Lemma 4, $M, \sigma \models \alpha'$. If $\alpha' = \alpha$ then we can derive from $M, \sigma \models \alpha'$ that $M, w \models A$. Suppose that $\alpha' \neq \alpha$. Thus, α is of the form $\triangle''\nabla_i\nabla'_i E$, where $\triangle''\nabla_i = \triangle$, $g(i)$ is a singleton, and ∇'_i is $\Box_i$ or $\langle E\rangle_i$. If $\nabla'_i = \langle E\rangle_i$ then $A = \Diamond_i E$. We have that $\alpha' = \triangle''\nabla'_i E$. Since $M, \sigma \models \alpha'$ and $g(i)$ is a singleton, it follows that $M, \sigma \models \triangle''\Box_i A$. Hence $M, w \models A$. This completes the proof.

Proof of Lemma 2 Let $\langle W, \tau, R_1, \ldots, R_m, H\rangle$ be the standard L-model graph of I, $\boxdot = \Box_{i_1} \ldots \Box_{i_k}$ be a modality, and $w = \langle\top\rangle_{i_1} \ldots \langle\top\rangle_{i_k}$.

– Consider the case α is of the form $\boxdot E$. Since $M \models \alpha$, we have $M, w \models E$. Hence, by Lemma 3, $\boxdot E \in Ext_L(I)$, and we also have $\boxdot E \in Sat_L(I)$.

– Now consider the case α is of the form $\boxdot\Diamond_i E$ with the property that if $g(i)$ is a singleton then $i \neq i_k$. Since $M \models \alpha$, we have $M, w \models \Diamond_i E$. Hence there exists u such that $R_i(w, u)$ holds and $E \in H(u)$.
It can be shown that u is of the form $w\langle F_1\rangle_{j_1} \ldots \langle F_h\rangle_{j_h}$ with $h \geq 1$ and $g(l) \subseteq g(i)$ for all $l \in \{j_1, \ldots, j_h\}$. (First, by induction on the derivation of $R_i(w, u)$, it can be shown that there are possible worlds $w_0, \ldots, w_n$ and indices $s_1, \ldots, s_n$ such that $w_0 = w$, $w_n = u$, and for every $1 \leq t \leq n$, either $R'_{s_t}(w_{t-1}, w_t)$ holds or $R_{s_t}(w_{t-1}, w_t)$ holds and $g(s_t)$ is a singleton. Observe that if $R_s(v, v')$ and $R_s(v', v'')$ hold then $R_s(v, v'')$ holds; if $R'_j(v, v')$ and $R_s(v', v'')$ hold, $j \neq s$, and $g(s)$ is a singleton, then $R'_s(v', v'')$ holds. Hence, we can assume that $R'_{s_t}(w_{t-1}, w_t)$ holds for all $1 \leq t \leq n$. Take $h = n$ and $j_t = s_t$ for $1 \leq t \leq n$.)
By Lemma 3, some $\Box$-lifting form of $w\langle F_1\rangle_{j_1} \ldots \langle F_h\rangle_{j_h} E$ belongs to $Ext_L(I)$. It follows that some $\Box$-lifting form of $\boxdot\langle F_1\rangle_{j_1} \ldots \langle F_h\rangle_{j_h} E$ belongs to $Ext_L(I)$ and $Sat_L(I)$. Hence $\boxdot\Diamond_i E$ is an L-instance of some atom from $Sat_L(I)$.

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

6 More Examples

In this section, we give two more examples illustrating the usefulness of modal logic programming for multi-agent systems.

6.1 Inheritance in a Hierarchy of Classes

In [5], Baldoni *et al.* formalizes an example taken from [6] of inheritance in a hierarchy of classes by a modal logic program. We adopt here their example with a small modification. Let us consider four classes: *animal*, *horse*, *bird*, and *tweety*. Since what is true for animals is also true for birds and horses, the *bird* and *horse* classes inherit from the *animal* class. Moreover, the class *tweety* inherits from *bird* and thus from *animal*.

Animals are not agents in the normal sense, but there is a similarity between the mentioned hierarchy of classes with a multi-agent system. We can treat the class *animal* as a group of agents, the class *bird* as its subgroup of agents, and so on. Program clauses with modal context *animal* can be applied for *horse*, *bird*, and *tweety*. Apparently, this feature is useful for defining epistemic states of groups of agents.

In the following, we use the mentioned classes as modal indices and write, e.g., $[animal]$ and $\langle animal\rangle$ respectively for $\Box_{animal}$ and $\Diamond_{animal}$ (and $\langle X\rangle_{animal}$ can be denoted by $\langle animal, X\rangle$). We use $KD4I_g5_a$, and for the hierarchy, we adopt the conditions that $g(animal) \supset g(horse)$, $g(animal) \supset g(bird)$, and $g(bird) \supset g(tweety)$. Furthermore, treating *tweety* as an object, we assume that $g(tweety)$ is a singleton.

As an example, we have the following clauses:

$$[animal]mode(walk).$$
$$[animal](mode(run) \leftarrow no\_of\_legs(X), X \geq 2).$$
$$[animal](mode(gallop) \leftarrow no\_of\_legs(X), X = 4).$$
$$[horse]no\_of\_legs(4).$$
$$[horse]covering(hair).$$
$$[bird]no\_of\_legs(2).$$
$$[bird]covering(feather).$$
$$[tweety]owner(fred).$$

The atom $[tweety]mode(run)$ can be derived from the above program in $KD4I_g5_a$. If the program contains also $[bird](mode(fly) \leftarrow light)$ or $[bird](mode(fly) \leftarrow \langle bird\rangle light)$, and $\langle tweety\rangle light$, then we can derive $\langle tweety\rangle mode(fly)$ (i.e. have a refutation for the goal $\leftarrow \langle tweety\rangle mode(fly)$). This and the example about the wise men puzzle demonstrate that using $KD4I_g5_a$-MProlog we can reason about possibility. This feature was not incorporated in the work [5] by Baldoni *et al.* (as they studied only modal logic programs without existential modal operators).

6.2 An Example for Modal Deductive Databases

For distributed systems of belief we can use the logic system

$$KD4_s5_s = K_m + (D) + (4_s) + (5_s)$$

where axioms (4_s): $\Box_i\varphi \rightarrow \Box_j\Box_i\varphi$ and (5_s): $\neg\Box_i\varphi \rightarrow \Box_j\neg\Box_i\varphi$ say that agents have full access to belief bases of each other. They are members of a united system and viewed as "friends". An SLD-resolution for MProlog in $KD4_s5_s$ is given in [23]. The following example is taken from our paper [27].

Let us consider the situation when a company has some branches and a central database. Each of the branches can access and update the database, and suppose that the company wants to distinguish data and knowledge coming from different branches. Also assume that data coming from branches can contain noises and statements expressed by a branch may not be highly recognized by other branches. This means that data and statements expressed by branches are treated as "belief" rather than "knowledge". In this case, we can use the multimodal logic $KD4_s5_s$, where each modal index represent a branch of the company, also called an *agent*. Recall that in this logic each agent has a full access to the belief bases of the other agents. Data put by agent i are of the form $\Box_i E$ (agent i believes in E) or $\Diamond_i E$ (agent i considers that E is possible). A statement expressed by agent i is a clause of the form $\Box_i(A \leftarrow B_1, \ldots, B_n)$, where A is an atom of the form E, $\Box_i E$, or $\Diamond_i E$, and $B_1, \ldots, B_n$ are simple modal atoms that may contain modal operators of the other agents. For communicating with normal users, the central database may contain rules with the empty modal context, i.e. in the form $E \leftarrow B_1, \ldots, B_n$, which hide sources of information. As

a concrete example, consider the following program/database in $KD4_s5_s$:

agent 1:

$\Box_1(\Diamond_1 likes(x, Coca) \leftarrow likes(x, Pepsi))$ (1)
$\Box_1(\Diamond_1 likes(x, Pepsi) \leftarrow likes(x, Coca))$ (2)
$\Box_1 likes(Tom, Coca) \leftarrow$ (3)
$\Box_1 likes(Peter, Pepsi) \leftarrow$ (4)

agent 2:

$\Box_2(likes(x, Coca) \leftarrow likes(x, Pepsi))$ (5)
$\Box_2(likes(x, Pepsi) \leftarrow likes(x, Coca))$ (6)
$\Box_2 likes(Tom, Pepsi) \leftarrow$ (7)
$\Box_2 likes(Peter, Coca) \leftarrow$ (8)
$\Box_2 likes(Peter, beer) \leftarrow$ (9)

agent 3:

$\Box_3(very\_much\_likes(x, y) \leftarrow likes(x, y), \Box_1 likes(x, y), \Box_2 likes(x, y))$ (10)
$\Box_3 likes(Tom, Coca) \leftarrow$ (11)
$\Diamond_3 likes(Peter, Pepsi) \leftarrow$ (12)
$\Diamond_3 likes(Peter, beer) \leftarrow$ (13)

for communicating with users:

$very\_much\_likes(x, y) \leftarrow \Box_3 very\_much\_likes(x, y)$ (14)
$likes(x, y) \leftarrow \Diamond_3 very\_much\_likes(x, y)$ (15)
$possibly\_likes(x, y) \leftarrow \Diamond_i likes(x, y)$ (for $i \in \{1, 2, 3\}$) (16)

In the above example, we assume that data and rules are stored in a central database. They can be stored also in a distributed database, where each agent (i.e. branch) has its own database. Such a distributed database can be treated as a multi-agent system.

7 Related Works and Possible Extensions

This paper considers only one of different aspects of multi-agent systems. In particular, we did not consider temporal dimension, actions, and events. Thus the current version of MProlog is not yet an agent programming language like AgentSpeak(L) [30], 3APL [16], and KARO [21]. In this work, we concentrated on reasoning about common/mutual belief, which was also considered in the paper [21] on KARO, but neglected in [31, 30, 16].

To deal with actions and time, possible solutions are to adopt CTL like the BDI-architecture [31], (concurrent) dynamic logic like the KARO system [21], or discrete linear temporal logic. Extending MProlog with dynamic logic or discrete linear temporal logic is possible, because such logics can be treated as modal logics. In our opinion, extending MProlog with concurrent dynamic logic is an interesting problem. Some temporal operators can be defined by modal operators of actions. Interaction between time and belief/knowledge is also a problem to be considered. For simplicity, one can study the case when temporal operators are outside the scope of belief/knowledge.

However, this is still not sufficient for practical multi-agent systems. There remain a lot of problems to be solved. In our opinion, multi-agent planning deserves for more attention. Also, perhaps we should use rewards and penalties for cooperative and competitive[5] multi-agent systems to deal with negotiation and cooperation. But in that case, it seems not easy to adopt logics for specification and verification of multi-agent systems.

Our related works are listed in the next section. Works involving with the wise men puzzle have been discussed in Section 3.

8 Conclusions

Our contributions in this paper are: the schema for semantics of $KD4I_g5_a$-MProlog given in Table 1, proofs of the soundness and completeness of SLD-resolution for $KD4I_g5_a$-MProlog, and a formalization of the wise men puzzle in the purely logical formalism of $KD4I_g5_a$-MProlog together with its SLD-refutation.

In this text, we recalled a large number of definitions and constructions from [26] (which in turn is an extension of [22]) in order to make the paper self-contained and understandable. Thus, the method used in this work for specifying and proving correctness of semantics of $KD4I_g5_a$-MProlog is not new. It originates or relates to our other works [22–26]. However, this does not reduce the originality of the above-mentioned contributions. They are first published in this paper.

The SLD-refutation given in Section 4.5 for the wise men puzzle does not uses rules or properties involving with axiom (5_a). Consequently, the puzzle can be solved in the logic $KD4I_g = K_m + (D) + (4) + (I_g)$. The choice of $KD4I_g5_a$ is justified as one of possible multimodal logics of belief and common/mutual belief that can be used to formalize the wise men puzzle. Our framework for modal logic programming [26] is applicable for a wide class of multimodal logics (see [23, 25]) and it can be extended for other versions of Kripke semantics, e.g. with varying domain or flexible terms (see a discussion in [26]).

In summary, this paper is on reasoning about common/mutual belief. It shows that the wise men puzzle can be nicely formalized in a multimodal logic of belief using modal logic programming. Our system is goal-driven and we focused on theoretical aspects like soundness and completeness. We did not incorporate actions and temporal dimension into our system, and this remains as an interesting problem.

Acknowledgements

I would like to thank the reviewers for helpful comments and suggestions.

[5] Environment can be treated as a competitive agent.

References

1. H. Aldewereld, W. van der Hoek, and J.-J.Ch. Meyer. Rational teams: Logical aspects of multi-agent systems. *Fundamenta Informaticae*, 63(2–3):159–183, 2004.
2. G. Attardi and M. Simi. Proofs in context. In J. Doyle, E. Sandewall, and P. Torasso, editors, *KR'94: Principles of Knowledge Representation and Reasoning*, pages 16–26, San Francisco, 1994. Morgan Kaufmann.
3. 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.
4. M. Baldoni. Normal multimodal logics with interaction axioms. In D. Basin, M. D'Agostino, D.M. Gabbay, and L. Viganò, editors, *Labelled Deduction*, pages 33–57. Kluwer Academic Publishers, 2000.
5. 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.
6. A. Brogi, E. Lamma, and P. Mello. Inheritance and hypothetical reasoning in logic programming. In *Proceedings of ECAI'90*, pages 105–110, Stockholm, 1990.
7. A. Cimatti and L. Serafini. Multi-agent reasoning with belief contexts: The approach and a case study. In M. Wooldridge and N.R. Jennings, editors, *Proceedings of ECAI-94, LNCS 890*, pages 71–85. Springer, 1995.
8. N. de Carvalho Ferreira, M. Fisher, and W. van der Hoek. Practical reasoning for uncertain agents. In J.J. Alferes and J.A. Leite, editors, *Proceedings of JELIA'2004*, volume 3229 of *LNCS*, pages 82–94. Springer-Verlag, 2004.
9. F. Debart, P. Enjalbert, and M. Lescot. Multimodal logic programming using equational and order-sorted logic. *Theoretical Comp. Science*, 105:141–166, 1992.
10. J.J. Elgot-Drapkin. Step-logic and the three-wise-men problem. In *AAAI*, pages 412–417, 1991.
11. R. Fagin, J.Y. Halpern, Y. Moses, and M.Y. Vardi. *Reasoning About Knowledge*. MIT Press, 1995.
12. L. Fariñas del Cerro. Molog: A system that extends Prolog with modal logic. *New Generation Computing*, 4:35–50, 1986.
13. M. Fisher and R. Owens. An introduction to executable modal and temporal logics. In M. Fisher and R. Owens, editors, *Executable Modal and Temporal Logics, IJCAI'93 workshop*, pages 1–20. Springer, 1995.
14. R. Goré and L.A. Nguyen. A tableau system with automaton-labelled formulae for regular grammar logics. In B. Beckert, editor, *Proceedings of TABLEAUX 2005, LNAI 3702*, pages 138–152. Springer-Verlag, 2005.
15. R. Goré and L.A. Nguyen. Tableaux for regular grammar logics of agents using automaton-modal formulae. To be submitted to JAR, 2006.
16. K.V. Hindriks, F.S. de Boer, W. van der Hoek, and J.-J.Ch. Meyer. Agent programming in 3APL. *Autonomous Agents and Multi-Agent Systems*, 2(4):357–401, 1999.
17. M. Kacprzak, A. Lomuscio, and W. Penczek. Bounded versus unbounded model checking for interpreted systems (invited talk at FAAMAS'03). In B. Dunin-Keplicz and R. Verbrugge, editors, *Proceedings of FAAMAS'03*, pages 5–20, 2003.
18. K. Konolige. Belief and incompleteness. Technical Report 319, SRI Inter., 1984.
19. J.W. Lloyd. *Foundations of Logic Programming, 2nd Ed.* Springer-Verlag, 1987.
20. J. McCarthy. First order theories of individual concepts and propositions. *Machine Intelligence*, 9:120–147, 1979.

21. J.-J.Ch. Meyer, F.S. de Boer, R.M. van Eijk, K.V. Hindriks, and W. van der Hoek. On programming KARO agents. *Logic Journal of the IGPL*, 9(2), 2001.
22. L.A. Nguyen. A fixpoint semantics and an SLD-resolution calculus for modal logic programs. *Fundamenta Informaticae*, 55(1):63–100, 2003.
23. 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.
24. 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.
25. L.A. Nguyen. An SLD-resolution calculus for basic serial multimodal logics. In D.V. Hung and M. Wirsing, editors, *Proceedings of ICTAC 2005, LNCS 3722*, pages 151–165. Springer, 2005.
26. L.A. Nguyen. The modal logic programming system MProlog: Theory, design, and implementation. Manuscript, available at `http://www.mimuw.edu.pl/~nguyen/mprolog`, 2005.
27. L.A. Nguyen. On modal deductive databases. In J. Eder, H.-M. Haav, A. Kalja, and J. Penjam, editors, *Proceedings of ADBIS 2005, LNCS 3631*, pages 43–57. Springer, 2005.
28. 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.
29. M.A. Orgun and W. Ma. An overview of temporal and modal logic programming. In D.M. Gabbay and H.J. Ohlbach, editors, *Proc. First Int. Conf. on Temporal Logic - LNAI 827*, pages 445–479. Springer-Verlag, 1994.
30. A.S. Rao. AgentSpeak(L): BDI agents speak out in a logical computable language. In *Proceedings of the 7th European Workshop MAAMAW*, volume 1038 of *LNCS*, pages 42–55. Springer, 1996.
31. A.S. Rao and M.P. Georgeff. Modeling rational agents within a BDI-architecture. In *KR*, pages 473–484, 1991.
32. R.A. Schmidt and D. Tishkovsky. Multi-agent logic of dynamic belief and knowledge. In S. Flesca, S. Greco, N. Leone, and G. Ianni, editors, *Proceedings of JELIA'2002*, volume 2424 of *LNAI*, pages 38–49. Springer, 2002.
33. W. van der Hoek and J.-J. Meyer. Modalities for reasoning about knowledge and uncertainties. In P. Doherty, editor, *Partiality, Modality, and Nonmonotonicity*. CSLI Publications, 1996.