Constructing Finite Least Kripke Models for Positive Logic Programs in Serial Regular Grammar Logics*

Linh Anh Nguyen

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

Abstract

A serial context-free grammar logic is a normal multimodal logic L characterized by the seriality axioms and a set of inclusion axioms of the form $\Box_t\varphi \rightarrow \Box_{s_1} \ldots \Box_{s_k}\varphi$. Such an inclusion axiom corresponds to the grammar rule $t \rightarrow s_1 \ldots s_k$. Thus the inclusion axioms of L capture a context-free grammar $\mathcal{G}(L)$. If for every modal index t, the set of words derivable from t using $\mathcal{G}(L)$ is a regular language, then L is a serial regular grammar logic.

In this paper, we present an algorithm that, given a positive multimodal logic program P and a set of finite automata specifying a serial regular grammar logic L, constructs a finite least L-model of P. (A model M is less than or equal to model M' if for every positive formula φ, if $M \models \varphi$ then $M' \models \varphi$.) A least L-model M of P has the property that for every positive formula φ, $P \models \varphi$ iff $M \models \varphi$. The algorithm runs in exponential time and returns a model with size $2^{O(n^3)}$. We give examples of P and L, for both of the case when L is fixed or P is fixed, such that every finite least L-model of P must have size $2^{\Omega(n)}$. We also prove that if G is a context-free grammar and L is the serial grammar logic corresponding to G then there exists a finite least L-model of $\Box_s p$ iff the set of words derivable from s using G is a regular language.

1 Introduction

Grammar logics were introduced by Fariñas del Cerro and Penttonen in [9] and have been studied widely [3, 7, 28, 13, 8, 10]. They are normal multimodal logics characterized by "inclusion" axioms like $\Box_{t_1} \ldots \Box_{t_h}\varphi \rightarrow \Box_{s_1} \ldots \Box_{s_k}\varphi$. Inclusion axioms correspond to grammar rules of the form $t_1 t_2 \ldots t_h \rightarrow s_1 s_2 \ldots s_k$ when modal indices are treated as grammar symbols. A grammar logic L is called a context-free grammar logic if the corresponding grammar $\mathcal{G}(L)$ is context-free, and is called a regular grammar logic if for every modal index t, the set of words derivable from t using $\mathcal{G}(L)$ is a regular language. A serial regular (resp. context-free) grammar logic is a regular (resp. context-free) grammar logic extended by the seriality axioms (i.e. $\Diamond_t\top$ for all modal indices t). These classes of serial multimodal logics contain useful epistemic logics. The seriality axioms (written in the form $\Box_i\varphi \rightarrow \neg\Box_i\neg\varphi$) state that knowledge and belief are consistent, while inclusion axioms can be used, for example, to express positive introspection

*Logic Journal of IGPL 2008 16: 175-193.

of knowledge and belief ($\Box_i\varphi \to \Box_i\Box_i\varphi$) or to represent knowledge sharing between agents or groups of agents (e.g. $\Box_i\varphi \to \Box_j\varphi$ or $\Box_i\varphi \to \Box_j\Box_i\varphi$).

Horn fragments of logics have received lot of attention because of the fact that logical implication in the form $B_1 \wedge \ldots \wedge B_k \to A$ is widely used in practice and the fact that by restricting to Horn fragments the computational complexity may be reduced in some cases. For example, Hustadt et al. [14] proved that the data complexity of the Horn-$\mathcal{SHIQ}$ fragment of the expressive description logic $\mathcal{SHIQ}$ is in PTIME, while the data complexity of $\mathcal{SHIQ}$ is coNP-complete. Since a positive logic program in a propositional modal logic can be used as a TBox in description logic for defining concepts, the study of Horn fragments of propositional modal logics is fully justifiable. Also note that the works on modal logic programming [1, 6, 25, 2, 20, 22] are based on Horn fragments of (first-order) modal logics, and the combination of description logics and Horn logic have recently been studied by a considerable number of researchers, e.g. [16, 4, 11, 5].

In this work, we study the Horn fragment of serial regular/context-free grammar logics. In particular, we study the problem of checking $P \models_L \varphi$ for a positive logic program P and a positive formula φ in a serial regular/context-free grammar logic L. Our method is bottom-up and based on constructing a finite least L-model M of P. (Kripke models are ordered by comparing the sets of positive consequences.) Thus, for any positive formula φ, $P \models_L \varphi$ iff $M \models \varphi$. As an example of application, we use this to formalize and solve the wise men puzzle. Our method is especially useful when P plays the role of a knowledge base that rarely changes, while φ is a query and varies. At least it seems more efficient than the usual tableau-based method when we have to check $P \models_L \varphi$ for a fixed positive logic program P and many positive formulae φ.

The problem of constructing a finite least L-model M of P is not trivial at all. When L is classical propositional logic (CPL), any model of P can be minimized into a least model of P, but the problem is not easy for modal logics. The reason is that a Kripke model has a structure which is not flat as in the case of CPL, and here we want to minimize the model w.r.t. the set of positive consequences but not w.r.t. the size. When L is a serial modal logic, if one translates the program P (together with the specification of the logic) into classical first-order logic using the functional translation [26, 6] or the semi-functional translation [25], then one obtains a Horn clause theory, which has a minimal Herbrand model. However, proving that this model can be collapsed into a finite model seems more complicated. Besides, converting a Herbrand model back into a Kripke model is hard in the case of the functional translation. Also note that the relational translation does not preserve the Horn property (e.g. applying the relational translation to the program clause $p \leftarrow \Box q$ results in $\forall x(p(x) \vee \exists y(R(x,y) \wedge \neg q(y)))$, where R is the accessibility relation). Furthermore, there are positive modal logic programs that do not have any least L-model, e.g. for $L = K$ or $L = K4$ (the problem of non-serial logics; see [19]). In some cases, for example $P = \{p\}$ and $L \in \{KDB, B\}$, a least L-model of P exists but it must be infinite.

In this paper, we present an algorithm that, given a positive multimodal logic program P and a set of finite automata specifying a serial regular grammar logic L, constructs a finite least L-model of P. The algorithm runs in exponential time and returns a model with size $2^{O(n^3)}$. We give examples of P and L, for both of the case when L is fixed or P is fixed, such that every finite least L-model of P must have size $2^{\Omega(n)}$. We also prove that if G is a context-free grammar and L is the serial grammar logic corresponding to G then there exists a finite least L-model of $\Box_s p$ iff the set of words derivable from s using G is a regular language.

This work is related to our previous work [19] and our joint work with Goré [10]. In [19] we gave an algorithm that for a given positive logic program P in a serial monomodal logic

$L \in \{KD,\ T,\ KDB,\ B,\ KD4,\ S4,\ KD5,\ KD45,\ S5\}$ constructs a least L-model of P, which is finite if $L \notin \{KDB, B\}$. When shifting to serial regular grammar logics, from the point of view of [19], the challenge is to manage to obtain the stop property (it is not easy when inclusion axioms are included) and exponential upper bound (instead of double exponential time and size). The solution is to use formulae with automaton-modal operators as in [10], which are similar to formulae of automaton propositional dynamic logic (APDL) [12]. In [10], we together with Goré used such formulae for developing analytic tableau calculi with the superformula property for regular grammar logics.[1] The technique used in the present work for constructing least L-models for the case when L is a serial regular grammar logic is a combination of the technique of [19], the use of automaton-modal operators [10, 12], and a special caching technique. It seems quite natural following [19, 10]. However, a more important matter is that with it, we obtain a much more significant result, as the class of serial regular grammar logics is large and contains useful epistemic logics, while the monomodal logics KD, T, $KD4$, $S4$ considered in [19] are just simple logics of this class.

The rest of this paper is structured as follows. In Section 1.1, we present the wise men puzzle as a motivational example. In Section 2, we define serial regular/context-free grammar logics, positive modal logic programs, an ordering of Kripke models, and formulae with automaton-modal operators. In Section 3, we present our algorithm for constructing finite least Kripke models of positive modal logic programs in serial regular grammar logics. Section 4 concerns complexity lower bounds of the considered problem. In Section 5, we show that a serial context-free grammar logic has the finite least Kripke model property for positive modal logic programs iff it is a serial regular grammar logic. Section 6 concludes this work.

1.1 A Motivational Example

The wise men puzzle is a famous benchmark introduced by McCarthy [18] for AI and has previously been studied in a considerable number of works (see [24] for some of them). The puzzle can be stated as follows (cf. [15]). 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 with a on front. 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!".

For $t \in \{a, b, c\}$, let $\Box_t\varphi$ stand for "the wise man t believes in φ", p_t stand for "the card of t is white", and q_t stand for "the card of t is black". Let g denote the group $\{a, b, c\}$ and let $\Box_g$ informally stand for a certain operator of "common belief" of the group g. Let L_{wmp} be the serial regular grammar logic with modal indices g, a, b, c and the following inclusion axioms:

$$\Box_g\varphi \rightarrow \Box_t\varphi \ \text{ and } \ \Box_t\varphi \rightarrow \Box_t\Box_t\varphi \ \text{ for } t \in \{g, a, b, c\}.$$

The wise men puzzle can be formalized as follows.

It is a common belief of the group that if y sits behind x then x's card is white whenever

[1] The superformula property is better known as the subformula property. It is just a matter of name. See [10] for a precise definition.

y considers this possible:

$$\Box_g\,(p_a \leftarrow \Diamond_b\, p_a)$$
$$\Box_g\,(p_a \leftarrow \Diamond_c\, p_a)$$
$$\Box_g\,(p_b \leftarrow \Diamond_c\, p_b)$$

The following clauses are "dual" to the above ones:

$$\Box_g\,(\Box_b\, q_a \leftarrow q_a)$$
$$\Box_g\,(\Box_c\, q_a \leftarrow q_a)$$
$$\Box_g\,(\Box_c\, q_b \leftarrow q_b)$$

It is a common belief of the group that at least one of the wise men has a white card:

$$\Box_g\,(p_a \leftarrow q_b, q_c)$$
$$\Box_g\,(p_b \leftarrow q_c, q_a)$$
$$\Box_g\,(p_c \leftarrow q_a, q_b)$$

It is a common belief of the group that: 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:

$$\Box_g \Diamond_b\, q_b$$
$$\Box_g \Diamond_c\, q_c$$

Let P_{wmp} be the "positive logic program" consisting of the above "program clauses".

The goal is to check whether wise man a believes that his card is white: that is, whether $\Box_a p_a$ is a logical consequence in L_{wmp} of P_{wmp}. We will continue this example in Figure 1.

2 Preliminaries

2.1 Definitions for Multimodal Logics

Our modal language is built from two disjoint sets: $\mathcal{MOD}$ is a finite set of modal indices and $\mathcal{PROP}$ is a set of primitive propositions. We use p for an element of $\mathcal{PROP}$ and use t and s for elements of $\mathcal{MOD}$. Formulae of our primitive language are recursively defined using the BNF grammar below:

$$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \rightarrow \varphi \mid \Box_t\varphi \mid \Diamond_t\varphi$$

A *Kripke frame* is a tuple $\langle W, \tau, (R_t)_{t\in\mathcal{MOD}}\rangle$, where W is a nonempty set of possible worlds, $\tau \in W$ is the actual world, and each R_t is a binary relation on W, called the accessibility relation for $\Box_t$ and $\Diamond_t$. If $R_t(w, u)$ holds then we say that the world u is accessible from the world w via R_t.

A *Kripke model* is a tuple $\langle W, \tau, (R_t)_{t\in\mathcal{MOD}}, h\rangle$, where $\langle W, \tau, (R_t)_{t\in\mathcal{MOD}}\rangle$ is a Kripke frame and h is a function mapping worlds to sets of primitive propositions. For $w \in W$, the set of primitive propositions "true" at w is $h(w)$.

A *model graph* is a tuple $\langle W, \tau, (R_t)_{t\in\mathcal{MOD}}, H\rangle$, where $\langle W, \tau, (R_t)_{t\in\mathcal{MOD}}\rangle$ is a Kripke frame and H is a function mapping worlds to formula sets.

Given a Kripke model $M = \langle W, \tau, (R_t)_{t\in\mathcal{MOD}}, h\rangle$ and a world $w \in W$, the *satisfaction relation* $\models$ is defined as usual for the classical connectives with two extra clauses for the modalities as below:

$$M, w \models \Box_t\varphi \quad \text{iff} \quad \forall v \in W.R_t(w, v) \text{ implies } M, v \models \varphi$$
$$M, w \models \Diamond_t\varphi \quad \text{iff} \quad \exists v \in W.R_t(w, v) \text{ and } M, v \models \varphi.$$

We say that φ *is true at* w *in* M if $M, w \models \varphi$. We say that φ *is true in* M and call M a *model of* φ if $M, \tau \models \varphi$.[2]

If we consider only Kripke models, with no restrictions on R_t, we obtain a normal multi-modal logic with a standard Hilbert-style axiomatization $K_{(m)}$.

2.2 Serial Regular Grammar Logics

Recall that a *finite automaton* A is a tuple $\langle \Sigma, Q, I, \delta, F \rangle$, where Σ is the alphabet (for our case, $\Sigma = \mathcal{MOD}$), Q is a finite set of states, $I \subseteq Q$ is the set of initial states, $\delta \subseteq Q \times \Sigma \times Q$ is the transition relation, and $F \subseteq Q$ is the set of accepting states. A *run* of A on a word $s_1 \dots s_k$ is a finite sequence of states $q_0, q_1, \dots, q_k$ such that $q_0 \in I$ and $\delta(q_{i-1}, s_i, q_i)$ holds for every $1 \leq i \leq k$. It is an *accepting run* if $q_k \in F$. We say that A *accepts* word w if there exists an accepting run of A on w. The set of all words accepted/recognized by A is denoted by $\mathcal{L}(A)$.

Given two binary relations R_1 and R_2 over W, their relational composition $R_1 \circ R_2 = \{(x, y) \mid \exists z \in W.\ R_1(x, z)\ \&\ R_2(z, y)\}$ is also a binary relation over W.

A *grammar logic* is a multimodal logic extending $K_{(m)}$ with "inclusion axioms" of the form $\Box_{t_1} \dots \Box_{t_h} \varphi \rightarrow \Box_{s_1} \dots \Box_{s_k} \varphi$, where $\{t_1, \dots t_h, s_1, \dots s_k\} \subseteq \mathcal{MOD}$. Each inclusion axiom corresponds to the frame restriction $R_{s_1} \circ \dots \circ R_{s_k} \subseteq R_{t_1} \circ \dots \circ R_{t_h}$ where the corresponding side stands for the identity relation if $k = 0$ or $h = 0$. A *serial grammar logic* is an extension of a grammar logic with the seriality axioms $\Diamond_t \top$ for all $t \in \mathcal{MOD}$. Each seriality axiom $\Diamond_t \top$ corresponds to the frame restriction $\forall x. \exists y. R_t(x, y)$. For a serial grammar logic L, the *L-frame restrictions* are the set of all the mentioned corresponding frame restrictions. A Kripke model is an *L-model* if its frame satisfies all the L-frame restrictions.

A formula φ is *L-satisfiable* if there exists an L-model of φ. Similarly, a formula set Γ is *L-satisfiable* if there exists an L-model of Γ (i.e. an L-model satisfying all the formulae of Γ). A formula φ is *L-valid* if it is true in all L-models. A formula φ is a *logical consequence in L* of Γ, write $\Gamma \models_L \varphi$, if every L-model of Γ is also a model of φ.

An inclusion axiom $\Box_{t_1} \dots \Box_{t_h} \varphi \rightarrow \Box_{s_1} \dots \Box_{s_k} \varphi$ can also be seen as the grammar rule $t_1 \dots t_h \rightarrow s_1 \dots s_k$ where the corresponding side stands for the empty word if $k = 0$ or $h = 0$. Thus the inclusion axioms of a (serial) grammar logic L capture a grammar $\mathcal{G}(L)$. $\mathcal{G}(L)$ is *context-free* if its rules are of the form $t \rightarrow s_1 \dots s_k$, and is *regular* if it is context-free and for every $t \in \mathcal{MOD}$ there exists a finite automaton A_t that recognizes the words derivable from t using $\mathcal{G}(L)$. If a context-free grammar over alphabet $\mathcal{MOD}$ contains a rule of the form $t \rightarrow s_1 \dots s_k$ then we call t a *variable* of the grammar, otherwise we call t a *terminal*. The alphabet $\mathcal{MOD}$ (of words recognized by the grammar) thus contains both variables and terminals. If t is a terminal then $A_t = \langle \mathcal{MOD}, \{0, 1\}, \{0\}, \{(0, t, 1)\}, \{1\} \rangle$.

A *serial regular* (resp. *context-free*) *grammar logic* L is a serial grammar logic whose inclusion axioms correspond to grammar rules that collectively capture a regular (resp. context-free) grammar $\mathcal{G}(L)$. A regular language is traditionally specified either by a regular expression or by a left/right linear grammar or by a finite automaton. The first two forms can be transformed in PTIME to an equivalent finite automaton that is at most polynomially larger. But checking whether a context-free grammar generates a regular language is undecidable (see, e.g., [17]). Hence, we cannot compute these automata if we are given an arbitrary serial regular grammar logic. We therefore assume that for each $t \in \mathcal{MOD}$ we are given an automaton

[2] As in modal logic programming (see, e.g., [22]), we treat a given modal logic program as local assumptions to the actual world. In description logics, however, a program representing the TBox is treated as global assumptions. As demonstrated in [21], our method works also for the second case.

A_t recognizing the words derivable from t using $\mathcal{G}(L)$. This is the *set of automata specifying L*.

Example 1 Let $\mathcal{MOD} = \{1, \ldots, m\}$ for a fixed m. Consider the serial grammar logic with the inclusion axioms $\Box_i\varphi \to \Box_j\Box_i\varphi$ for any $i, j \in \mathcal{MOD}$ and $\Box_i\varphi \to \Box_j\varphi$ if $i > j$. This is a serial regular grammar logic because the set of words derivable from i using the corresponding grammar is represented by $(1 \mid \ldots \mid m)^*(1 \mid \ldots \mid i)$. For each i, the set is recognized by the automaton $A_i = \langle \mathcal{MOD}, \{p, q\}, \{p\}, \delta_i, \{q\}\rangle$ with $\delta_i = \{(p, j, p) \mid 1 \leq j \leq m\} \cup \{(p, j, q) \mid 1 \leq j \leq i\}$.

Example 2 Let $A = \langle \Sigma, Q, \{q_I\}, \delta, F\rangle$, where $q_I \in Q$ and $\delta : Q \times \Sigma \to Q$, be a deterministic finite automaton. Let G be the standard right-linear regular grammar generating the language recognized by A, i.e. G has alphabet Σ, variables X_q for $q \in Q$, the start symbol X_{q_I}, and rules $X_q \to \sigma X_{\delta(q,\sigma)}$ for $\sigma \in \Sigma$, $X_q \to \varepsilon$ for $q \in F$. Extending the alphabet to $\Sigma \cup Q$ and treating X_q as q, we obtain from G a regular grammar G' over the extended alphabet. This regular grammar corresponds to the serial regular grammar logic specified by $\mathcal{MOD} = Q \cup \Sigma$ and the inclusion axioms $\Box_q\varphi \to \Box_\sigma\Box_{\delta(q,\sigma)}\varphi$ for $q \in Q$ and $\sigma \in \Sigma$, and $\Box_q\varphi \to \varphi$ for $q \in F$. The set of automata specifying this serial regular grammar logic (and corresponding to G') consists of the automata $A_q = \langle Q \cup \Sigma, Q \cup \{f\}, \{q\}, \delta_q, F \cup \{f\}\rangle$ for $q \in Q$, where $f \notin Q$ and $\delta_q = \delta \cup \{(p, p, f) \mid p \in Q\}$, and $A_\sigma = \langle Q \cup \Sigma, \{0, 1\}, \{0\}, \{(0, \sigma, 1)\}, \{1\}\rangle$ for $\sigma \in \Sigma$.

Lemma 1 (cf. [3, 7, 10]) *Let L be a serial context-free grammar logic. Then the following conditions are equivalent:*

(1) *$s_1 \ldots s_k$ is derivable from t using the grammar $\mathcal{G}(L)$,*

(2) *the formula $\Box_t\varphi \to \Box_{s_1} \ldots \Box_{s_k}\varphi$ is L-valid,*

(3) *the inclusion $R_{s_1} \circ \cdots \circ R_{s_k} \subseteq R_t$ is a consequence of the L-frame restrictions.*

Proof. The implication (1) $\Rightarrow$ (2) follows by induction on the length of the derivation of $s_1 \ldots s_k$ from t by the grammar $\mathcal{G}(L)$, using substitution, the K-axiom $\Box_t(\varphi \to \psi) \to (\Box_t\varphi \to \Box_t\psi)$ and the modal necessitation rule $\varphi/\Box_t\varphi$. The equivalence (2) $\Leftrightarrow$ (3) is well-known from correspondence theory [27]. The implication (3) $\Rightarrow$ (1) follows by induction on the length of the derivation of $R_{s_1} \circ \ldots \circ R_{s_k} \subseteq R_t$ from the L-frame restrictions (for this, first observe that the conditions of seriality are not essential for the derivation of $R_{s_1} \circ \cdots \circ R_{s_k} \subseteq R_t$ from the L-frame restrictions).[3] •

2.3 Positive Modal Logic Programs

A *program clause* is a formula of the form $\boxdot(B_1 \wedge \ldots \wedge B_k \to A)$, where $\boxdot$ is a (possibly empty) sequence of universal modal operators (i.e. $\Box_t$), $k \geq 0$, and $B_1, \ldots, B_k, A$ are simple modal atoms of the form p, $\Box_s p$, or $\Diamond_s p$. We often write program clauses in the form $\boxdot(A \leftarrow B_1, \ldots, B_k)$, where $\boxdot$ is called the modal context, A the head, and $B_1, \ldots, B_k$ the body of the program clause.

A *positive (multimodal) logic program* is a finite set of program clauses.

A formula is in *negation normal form* if it does not contain the connective $\to$ and each negation occurs immediately before a primitive proposition. Every formula can be transformed to its equivalent negation normal form in the usual way. A formula is called *positive* if its

[3] The conditions of seriality do not play any role in this proof. They are needed somewhere else to guarantee the existence of least Kripke models of positive logic programs.

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

Horn formulae are recursively defined as follows: a primitive proposition is a Horn formula; a negative formula is a Horn formula; if φ and ψ are Horn formulae and ξ is a negative formula then $\Box_t\varphi$, $\Diamond_t\varphi$, $\varphi \wedge \psi$, and $\varphi \vee \xi$ are Horn formulae.

It can be shown that for every set X of Horn formulae, there exists a positive logic program P and a set Q of negative formulae such that X is L-satisfiable iff $P \cup Q$ is L-satisfiable (see [19] for the technique). Note that $P \cup Q$ is L-satisfiable iff $P \nvdash_L \varphi$, where φ is the negation of the conjunction of the formulae of Q (and thus a positive formula). As we will show, if L is a serial regular grammar logic, then P has a "least L-model" M such that $P \models_L \psi$ iff $M \models \psi$, for every positive formula ψ.

2.4 Ordering Kripke Models

A model M is said to be *less than* or *equal to* M', write $M \leq M'$, if for any positive formula φ, if M satisfies φ then M' also satisfies φ. This relation $\leq$ is a pre-order[4]. We write $M \equiv M'$ to denote that $M \leq M'$ and $M' \leq M$.

An L-model is a least L-model of a positive logic program P if it is an L-model of P and is less than or equal to every L-model of P.

Note that $M \equiv M'$ does not mean that $M = M'$. In particular, if M and M' are least L-models of P then $M \equiv M'$ but we do not have that $M = M'$. The equivalence $M \equiv M'$ only says that for every positive formula φ, $M \models \varphi$ iff $M' \models \varphi$.

Let $M = \langle W, \tau, (R_t)_{t \in \mathcal{MOD}}, h\rangle$ and $M' = \langle W', \tau', (R'_t)_{t \in \mathcal{MOD}}, h'\rangle$ be Kripke models. We say that M is *less than or equal to M' w.r.t. a binary relation* $r \subseteq W \times W'$, and write $M \leq_r M'$, if the following conditions hold for every $t \in \mathcal{MOD}$, $x \in W$ and $x' \in W'$:

1. $r(\tau, \tau')$
2. $\forall\, y \in W\ (R_t(x,y) \wedge r(x,x') \rightarrow \exists y' \in W'\,(R'_t(x',y') \wedge r(y,y')))$
3. $\forall\, y' \in W'\ (R'_t(x',y') \wedge r(x,x') \rightarrow \exists y \in W\,(R_t(x,y) \wedge r(y,y')))$
4. $r(x,x') \rightarrow h(x) \subseteq h(x')$

In the above definition, the first three conditions state that r is a kind of bisimulation of the *frames* of M and M'. (If we replace $h(x) \subseteq h(x')$ in the last condition by $h(x) = h(x')$ then r will be a bisimulation of M and M'.) Intuitively, $r(x,x')$ states that the world x is less than or equal to x'.

Lemma 2 *If $M \leq_r M'$ then $M \leq M'$.*

The order $\leq_r$ was introduced in our previous work [19], in which we proved the above lemma for normal monomodal logics. The proof for normal multimodal logics is similar (i.e. by induction on the length of φ that, if $M, w \models \varphi$ then $M', w \models \varphi$).

We give below a converse of the above lemma.

If $M = \langle W, \tau, (R_t)_{t \in \mathcal{MOD}}, h\rangle$ and $w \in W$ then by (M, w) we denote the Kripke model obtained from M by using w as the actual world.

A Kripke model $M = \langle W, \tau, (R_t)_{t \in \mathcal{MOD}}, h\rangle$ is said to be *finitely branching* if for every $u \in W$ and every $t \in \mathcal{MOD}$, the set $\{v \mid R_t(u,v)\}$ is finite.

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

Lemma 3 *Let M and M' be finitely branching Kripke models such that $M \leq M'$. Then $M \leq_r M'$ for some r.*

Proof. Let $M = \langle W, \tau, (R_t)_{t\in\mathcal{MOD}}, h\rangle$ and $M' = \langle W', \tau', (R'_t)_{t\in\mathcal{MOD}}, h'\rangle$. Define $r = \{(w, w') \mid w \in W,\ w' \in W',\ (M, w) \leq (M', w')\}$. We prove that $M \leq_r M'$. Clearly, $r(\tau, \tau')$ holds, and if $r(x, x')$ holds then $h(x) \subseteq h(x')$.

Suppose that $R_t(x, y)$ and $r(x, x')$ hold. We show that there exists $y' \in W'$ such that $R'_t(x', y')$ and $r(y, y')$ hold. For the sake of contradiction, suppose that for every $y' \in W'$ such that $R'_t(x', y')$ holds, $r(y, y')$ does not hold, i.e. there exists a positive formula $\varphi_{y'}$ such that $M, y \models \varphi_{y'}$ but $M', y' \not\models \varphi_{y'}$. For every $y' \in W'$ such that $R'_t(x', y')$ holds, choose such a $\varphi_{y'}$. Then because M' is finitely branching, one can construct a finite conjunction φ from the chosen $\varphi_{y'}$. We have $M, x \models \Diamond_t\varphi$, while $M', x' \not\models \Diamond_t\varphi$, which contradicts the assumption that $r(x, x')$ holds.

Suppose that $R'_t(x', y')$ and $r(x, x')$ hold. We show that there exists $y \in W$ such that $R_t(x, y)$ and $r(y, y')$ hold. For the sake of contradiction, suppose that for every $y \in W$ such that $R_t(x, y)$ holds, $r(y, y')$ does not hold, i.e. there exists a positive formula φ_y such that $M, y \models \varphi_y$ but $M', y' \not\models \varphi_y$. For every $y \in W$ such that $R_t(x, y)$ holds, choose such a φ_y. Then because M is finitely branching, one can construct a finite disjunction φ from the chosen φ_y. We have $M, x \models \Box_t\varphi$, while $M', x' \not\models \Box_t\varphi$, which contradicts the assumption that $r(x, x')$ holds. This completes the proof. •

2.5 Automaton-Modal Operators

If A is a finite automaton, Q is a subset of the states of A, and φ is a formula in the primitive language then we call $[A, Q]$ a (universal) *automaton-modal operator* and $[A, Q]\varphi$ a formula (in the extended language). In [10], $[A, Q]\varphi$ is denoted by $(A, Q) : \varphi$ and called an *automaton-labeled formula.*

Fix a serial regular grammar logic L and let $(A_t = \langle\mathcal{MOD}, Q_t, I_t, \delta_t, F_t\rangle)_{t\in\mathcal{MOD}}$ be the automata specifying L. Let $\delta_t(Q, s) = \{q' \mid \exists q \in Q.(q, s, q') \in \delta_t\}$ be the states which can be reached from Q via an s-transition using A_t. Let ε be the empty word and define $\widetilde{\delta_t}(Q, \varepsilon) = Q$ and $\widetilde{\delta_t}(Q, s_1 \dots s_k) = \delta_t(\widetilde{\delta_t}(Q, s_1 \dots s_{k-1}), s_k)$.

The formal semantics of formulae with automaton-modal operators is defined as follows: $M, w_0 \models [A_t, Q]\varphi$ if $M, w_k \models \varphi$ for every path $w_0 R_{s_1} w_1 \dots w_{k-1} R_{s_k} w_k$ with $k \geq 0$ such that $\widetilde{\delta_t}(Q, s_1 \dots s_k) \cap F_t \neq \emptyset$ (i.e. $s_1 \dots s_k$ is accepted by A_t when starting from some state from Q).

It can be easily seen that formulae with automaton-modal operators satisfy the following reasoning rules:

- A formula of the form $\Box_t\varphi$ at a world u is represented by $[A_t, I_t]\varphi$.
- If $[A_t, Q]\varphi$ occurs at u and $R_s(u, v)$ holds then we add the formula $[A_t, \delta_t(Q, s)]\varphi$ to v. In particular, if $[A_t, I_t]\varphi$ appears in world u and $R_s(u, v)$ holds then we add $[A_t, \delta_t(I_t, s)]\varphi$ to the world v.
- If $[A_t, Q]\varphi$ and $[A_t, Q']\varphi$ occur at u then we replace them by $[A_t, Q \cup Q']\varphi$.
- If $[A_t, Q]\varphi$ occurs at u and Q contains an accepting state of A_t, then we add φ to u.

Our formulae with automaton-modal operators are similar to formulae of automaton propositional dynamic logic (APDL) [12]. A formula involving automata in APDL is of the form

$[A]\varphi$, where A is a finite automaton with *one* initial state and *one* accepting state. A formula like our $[A_t, Q]\varphi$ with $Q = \{q_1, q_2, \ldots, q_k\}$ can be simulated by the APDL formula $[B_1]\varphi \vee \ldots \vee [B_k]\varphi$ where each B_i is an automaton with one accepting state equivalent to the automaton A_t restricted to start at the initial state q_i. Thus our formulation uses a more compact representation in which APDL formulae that differ only in their initial state are grouped together.

From now on, by a *formula* we mean a formula in the extended language.

3 Constructing Finite Least Kripke Models

In this section, we present an algorithm that, given a positive logic program P and a serial regular grammar logic L specified by a set of finite automata, constructs a finite least L-model of P. The seriality axioms are needed to guarantee the existence of least Kripke models of positive logic programs. Recall that there are positive modal logic programs that do not have any least Kripke model in the non-serial modal logics K and $K4$ [19].

Let X be a set of formulae. The saturation of X, denoted by $\mathsf{Sat}(X)$, is defined to be the least extension of X such that:

- if $\Box_t\varphi \in \mathsf{Sat}(X)$ then $[A_t, I_t]\varphi \in \mathsf{Sat}(X)$,
- if $[A_t, Q]\varphi \in \mathsf{Sat}(X)$ and $Q \cap F_t \neq \emptyset$ then $\varphi \in \mathsf{Sat}(X)$,
- $\top \in \mathsf{Sat}(X)$ and $\Diamond_t\top \in \mathsf{Sat}(X)$ for every $t \in \mathcal{MOD}$.

The transfer of X through $\Diamond_t$, denoted by $\mathsf{Trans}(X, t)$, is defined to be $\mathsf{Sat}(\{[A_s, \delta_s(Q, t)]\psi \mid [A_s, Q]\psi \in X\})$.

The compact form of X, denoted by $\mathsf{CF}(X)$, is the least set of formulae obtained as follows:

- if $\varphi \in X$ and φ is not of the form $[A_t, Q]\psi$ then $\varphi \in \mathsf{CF}(X)$,
- if $[A_t, Q]\psi \in X$ and $Q_1, \ldots, Q_k$ are all the sets such that $[A_t, Q_i]\psi \in X$ for $1 \leq i \leq k$, then $[A_t, Q_1 \cup \ldots \cup Q_k]\psi \in \mathsf{CF}(X)$.

The algorithm given below will construct the following data structures:

- W : a set of possible worlds, where $\tau \in W$ is the actual world.
- H : for every $w \in W$, $H(w)$ is a set of formulae called the content of w.
- $Next : W \times (\{\Diamond_t\top \mid t \in \mathcal{MOD}\} \cup \{\Diamond_t p \mid t \in \mathcal{MOD}, p \in \mathcal{PROP}\}) \rightarrow W$, a partial function which has the following intuitive meaning: $Next(u, \Diamond_t\varphi) = v$ means $\Diamond_t\varphi \in H(u)$, $\varphi \in H(v)$, and $\Diamond_t\varphi$ is "realized" at u by going to v.

Using the above data structures, we define:

- h to be the restriction of H such that $h(u) = H(u) \cap \mathcal{PROP}$ for $u \in W$;
- R'_t to be the accessibility relation on W such that $R'_t(u, v)$ holds if $Next(u, \Diamond_t\varphi) = v$ for some φ;
- R_t to be the least extension of R'_t such that $(R_t)_{t \in \mathcal{MOD}}$ satisfies all the L-frame restrictions except the seriality conditions;

- $M = \langle W, \tau, (R_t)_{t \in \mathcal{MOD}}, h \rangle$.

In the algorithm given below, we use the procedure $\mathsf{Find}(X, W, H)$ defined as: if there exists a world $u \in W$ with $H(u) = X$ then return u, else add a new world u to W with $H(u) = X$ and return u.

A finite least L-model of P is constructed by building an L-model graph of P. At the beginning the model graph contains only one world with content P. Then for every world u and every formula φ belonging to the content of u, if φ is not true at u then the algorithm makes a change to satisfy it. There are three main forms for $\varphi : A \leftarrow B_1, \ldots, B_k$ with $k \geq 1$, $[A_t, Q]\psi$, and $\Diamond_t \psi$ (the form $\Box_t \psi$ is reduced to $[A_t, I_t]\psi$). For the case of $A \leftarrow B_1, \ldots, B_k$, if all $B_1, \ldots, B_k$ are true and will remain true[5] at u then we would like to add A to the content of u. But if we do so then this may result in the situation that $H(u) = H(u')$ for some $u' \neq u$. To restrict the size of the constructed model graph, we prevent that situation as follows. Instead of adding A to the content of u, we redirect the connections to u to a world with an appropriate content, which is created if necessary. That is we use another world with an appropriate content to "replace" the role of u. The world u is not deleted, as we want to cache all worlds appearing during the process in order to increase efficiency. For the case when φ is of the form $[A_t, Q]\psi$, we would like to add $[A_t, \delta_t(Q, s)]\psi$ to the content of every world w accessible directly from u via R'_s, for $s \in \mathcal{MOD}$. But modifying the content of w may affect the other worlds connected to w. For example, if $R_i(v, w)$ holds then adding p to $H(w)$ causes $\Diamond_i p$ true at v. So, analogously as for the previous case, instead of modifying contents of worlds, we just redirect connections appropriately. For the case when φ is of the form $\Diamond_t \psi$, to satisfy φ at u, we connect u via R'_t to the world with content consisting of ψ and the formulae "inherited" from u via R'_t. To guarantee the constructed model graph to be smallest, we add $\Diamond_t \top$, for $t \in \mathcal{MOD}$, to the content of every world. Adding $\Diamond_t \top$ to the content of u causes u be connected to a "minimal" world via R_t. It also guarantees R_t to be serial at u.

Algorithm 1

Input: A positive logic program P and a serial regular grammar logic L specified by a set of finite automata $A_t = \langle \mathcal{MOD}, Q_t, I_t, \delta_t, F_t \rangle$ for $t \in \mathcal{MOD}$.

Output: $M = \langle W, \tau, (R_t)_{t \in \mathcal{MOD}}, h \rangle$ – a finite least L-model of P.

1. $W := \{\tau\}$; $H(\tau) := \mathsf{CF}(\mathsf{Sat}(P))$;
2. for every $u \in W$ and every $\varphi \in H(u)$
   (a) case $\varphi = A \leftarrow B_1, \ldots, B_k$ for some $k \geq 1$: if for every $1 \leq i \leq k$, $M, u \models B_i$ and if $B_i = \Box_t p$ then $Next(u, \Diamond_t \top)$ is defined, then
      i. $u_* := \mathsf{Find}(\mathsf{CF}(H(u) \cup \mathsf{Sat}(\{A\})), W, H)$;
      ii. for every $v \in W$, $t \in \mathcal{MOD}$, and $\psi \in \{\top\} \cup \mathcal{PROP}$, if $Next(v, \Diamond_t \psi) = u$ then $Next(v, \Diamond_t \psi) := u_*$;
      iii. if $\tau = u$ then $\tau := u_*$;
   (b) case $\varphi = [A_t, Q]\psi$: for every $w \in W$ and $s \in \mathcal{MOD}$ s.t. $R'_s(u, w)$ holds:
      i. $w_* := \mathsf{Find}(\mathsf{CF}(H(w) \cup \mathsf{Trans}(\{\varphi\}, s)), W, H)$;

[5] Observe that if at some step w is the only world accessible from u via R_t and $p \in H(w)$ then $M, u \models \Box_t p$, but this does not mean that $\Box_t p$ follows from the content of u. The truth of p or $\Diamond_t p$ at u can be checked in the usual way, but $\Box_t p$ is true and will remain true at u only if p is true at the world $Next(u, \Diamond_t \top)$.

ii. for every $\xi \in \{\top\} \cup \mathcal{PROP}$,
if $Next(u, \Diamond_s \xi) = w$ then $Next(u, \Diamond_s \xi) := w_*$;

(c) case $\varphi = \Diamond_t \psi$ (for $\psi \in \{\top\} \cup \mathcal{PROP}$):
if $Next(u, \Diamond_t \psi)$ is not defined then
$Next(u, \Diamond_t \psi) := \mathsf{Find}(\mathsf{CF}(\mathsf{Trans}(H(u), t)) \cup \{\psi\}, W, H)$;

3. while some change occurred, repeat step 2.

Note that in the step 2a, if $u_* \neq u$ then after executing the steps 2(a)i–2(a)iii, the world u is not reachable from τ (via any path using the accessibility relations R'_t, $t \in \mathcal{MOD}$).

As example, in Figure 1 we apply the above algorithm to the wise men puzzle.

Proposition 4 *Algorithm 1 terminates in $2^{O(n^3)}$ steps and returns a Kripke model with $2^{O(n^3)}$ worlds, where n is the size of input (i.e. the sum of the lengths of the clauses of P, the size of $\mathcal{MOD}$, and the sizes of the automata specifying L).*

Proof. For each $u \in W$ and $\varphi \in H(u)$, φ is either a subformula of a clause of P or a formula of the form $[A_t, Q]\psi$ with ψ being a subformula of a clause of P. There are less than n possible values for t, less than 2^n possible values for Q, and less than n possible values for ψ. Hence, due to the compact form, there are no more than $2^{O(n^3)}$ possible values for $H(u)$. Since the worlds of W have different contents, the size of W is $2^{O(n^3)}$. Also note that the worlds of W are never deleted and their contents do not change.

The step 2c makes a change no more than $2^{O(n^3)}.n.n = 2^{O(n^3)}$ times. For the steps 2a and 2b, note that the content of u_* (resp. w_*) is "bigger" than the content of u (resp. w). Hence $Next$ is modified by the steps 2a or 2b no more than $2^{O(n^3)}.n.n.2^{O(n^3)} = 2^{O(n^3)}$ times, and τ is modified no more than $2^{O(n^3)}$ times.

Therefore, we conclude that Algorithm 1 terminates in $2^{O(n^3)}$ steps and returns a Kripke model with $2^{O(n^3)}$ worlds, where each world is of size $O(n)$. •

Lemma 5 *Consider a moment in an execution of Algorithm 1. Suppose that $R_t(u, w)$ holds. Then there exist $w_0, \ldots, w_k$ in W with $w_0 = u$, $w_k = w$, and indices $s_1, \ldots, s_k \in \mathcal{MOD}$ such that $R'_{s_i}(w_{i-1}, w_i)$ holds for $1 \leq i \leq k$, and $R_{s_1} \circ \ldots \circ R_{s_k} \subseteq R_t$ follows from the L-frame restrictions.*

Proof. By induction on the number of inferences in the derivation of $R_t(u, w)$ when extending R'_s to R_s for $s \in \mathcal{MOD}$ using the L-frame restrictions. •

Lemma 6 *Let P be a positive logic program, L be a serial regular grammar logic, and M be the model returned by Algorithm 1 for P and L. Then M is an L-model of P.*

Proof. We will refer to the data structures used in Algorithm 1. It is clear that M is an L-model. To prove that $M \models P$, we show that for every $u \in W$ reachable from τ via a path using the accessibility relations R'_t and for every formula $\varphi \in H(u)$ without automaton-modal operators, $M, u \models \varphi$. We prove this by induction on the structure of φ. The case when $\varphi = \Diamond_t \psi$ is trivial.

Consider the case when $\varphi = A \leftarrow B_1, \ldots, B_k$ and the steps 2(a)ii and 2(a)iii are executed. As no changes occur (at the end) and u is reachable from τ via a path using the accessibility relations R'_t, we have that $u_* = u$. Thus, by the inductive assumption, $M, u \models A$, and hence $M, u \models \varphi$.

In this figure, we continue the example about the wise men puzzle. Recall that the program used for formalizing the puzzle is $P_{wmp} = \{\Box_g\varphi_1, \ldots, \Box_g\varphi_{11}\}$, where

$$\begin{array}{ll}
\varphi_1 = (p_a \leftarrow \Diamond_b\, p_a) & \varphi_7 = (p_a \leftarrow q_b, q_c) \\
\varphi_2 = (p_a \leftarrow \Diamond_c\, p_a) & \varphi_8 = (p_b \leftarrow q_c, q_a) \\
\varphi_3 = (p_b \leftarrow \Diamond_c\, p_b) & \varphi_9 = (p_c \leftarrow q_a, q_b) \\
\varphi_4 = (\Box_b\, q_a \leftarrow q_a) & \varphi_{10} = \Diamond_b\, q_b \\
\varphi_5 = (\Box_c\, q_a \leftarrow q_a) & \varphi_{11} = \Diamond_c\, q_c \\
\varphi_6 = (\Box_c\, q_b \leftarrow q_b) &
\end{array}$$

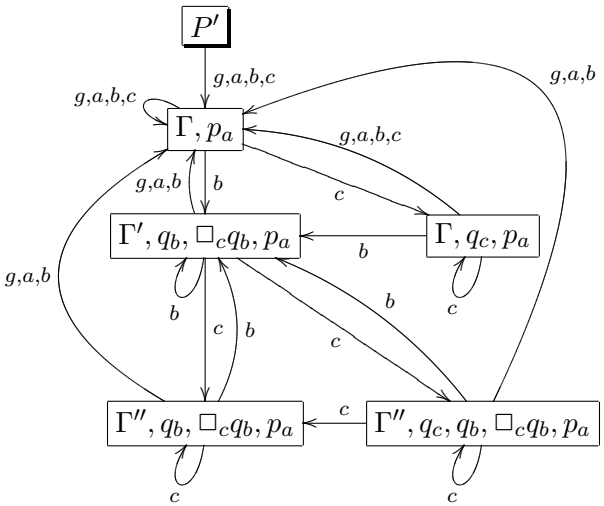
The used logic L_{wmp} has $\mathcal{MOD} = \{g, a, b, c\}$ and is specified by the set of the following automata:

$$A_g = \langle \mathcal{MOD}, \{0,1\}, \{0\}, \{(0,t,0), (0,t,1) \mid t \in \mathcal{MOD}\}, \{1\}\rangle,$$
$$\text{and for } t \in \{a,b,c\} : \; A_t = \langle \mathcal{MOD}, \{0,1\}, \{0\}, \{(0,t,1)\}, \{1\}\rangle.$$

Let

$$\begin{array}{lcl}
P' & = & \mathsf{Sat}(P) = P \cup \{[A_g, \{0\}]\varphi_i \mid 1 \le i \le 11\} \cup \{\top\} \cup \{\Diamond_t \top \mid t \in \mathcal{MOD}\}, \\
\Gamma & = & \{[A_g, \{0,1\}]\varphi_i,\, \varphi_i \mid 1 \le i \le 11\} \cup \{\top\} \cup \{\Diamond_t \top \mid t \in \mathcal{MOD}\}, \\
\Gamma' & = & \Gamma \cup \{[A_c, \{0\}]q_b\}, \\
\Gamma'' & = & \Gamma \cup \{[A_c, \{0,1\}]q_b\}.
\end{array}$$

We give below the model graph created by Algorithm 1 for P_{wmp} and L_{wmp}. The initial node τ is the node with a shaded frame. In the graph, an edge from a node u to a node w with a label $t \in \mathcal{MOD}$ means an edge from u to w via R'_t. The formula p_a is added to the right bottom node due to φ_7, and after that it is added to the other nodes except τ due to φ_1 or φ_2. We do not show in the graph the nodes that are not directly nor indirectly accessible from the initial node τ.



The formula $\Box_a p_a$ is true in the corresponding L_{wmp}-model of the model graph. By Theorem 9, which is presented and proved in the follows, $\Box_a p_a$ is a logical consequence in L_{wmp} of P_{wmp}.

Figure 1: An application of Algorithm 1 to the wise men puzzle

Consider the case when $\varphi = \Box_t\psi$. Suppose that $R_t(u,w)$ holds. By the inductive assumption, it is sufficient to show that $\psi \in H(w)$. Since $R_t(u,w)$ holds, by Lemma 5, there exist $w_0,\ldots,w_k$ in W with $w_0 = u$, $w_k = w$, and indices $s_1,\ldots,s_k \in \mathcal{MOD}$ such that $R'_{s_i}(w_{i-1},w_i)$ holds for $1 \le i \le k$, and $R_{s_1} \circ \ldots \circ R_{s_k} \subseteq R_t$ follows from the L-frame restrictions. By Lemma 1, $s_1 \ldots s_k$ is accepted by A_t. Hence $\widetilde{\delta}_t(I_t, s_1 \ldots s_k) \cap F_t \neq \emptyset$. Since $\varphi \in H(u)$, we have $[A_t, Q]\psi \in H(w_0)$ for some $Q \supseteq I_t$, and hence $[A_t, Q']\psi \in H(w_k)$ for some $Q' \supseteq \widetilde{\delta}_t(I_t, s_1 \ldots s_k)$. It follows that $\psi \in H(w_k)$, which means $\psi \in H(w)$. •

Lemma 7 *Let P be a positive logic program, L be a serial regular grammar logic, and $M' = \langle W', \tau', (S_t)_{t\in\mathcal{MOD}}, h'\rangle$ be an arbitrary L-model of P. Consider a moment after executing a numerated step in an execution of Algorithm 1 for P and L. Let $r = \{(x,x') \in W \times W' \mid M', x' \models H(x)\}$. (Here, W and H are the data structures used in Algorithm 1.) Then:*

a) $r(\tau,\tau')$ holds,

b) for every $x,y \in W$, $x',y' \in W'$, $l \in \mathcal{MOD}$, $\psi \in \{\top\} \cup \mathcal{PROP}$, if $r(x,x')$ holds, $Next(x, \Diamond_l\psi) = y$, $S_l(x',y')$ holds, and $M', y' \models \psi$, then $r(y,y')$ holds.

Proof. By induction on the number of executed steps.

The base case occurs after executing step 1 and the assertions clearly hold.

Consider some latter step of the algorithm. As induction hypothesis, assume that the assertions hold before executing that step. Suppose that after executing the step we have M_2, W_2, H_2, $Next_2$, $R'_{2,t}$, $R_{2,t}$ (for $t \in \mathcal{MOD}$), and r_2 in the places of M, W, H, $Next$, R'_t, R_t, and r. We prove that, a) $r_2(\tau,\tau')$ holds, b) for every $x,y \in W_2$, $x',y' \in W'$, $l \in \mathcal{MOD}$, $\psi \in \{\top\} \cup \mathcal{PROP}$, if $r_2(x,x')$ holds, $Next_2(x, \Diamond_l\psi) = y$, $S_l(x',y')$ holds, and $M', y' \models \psi$, then $r_2(y,y')$ holds.

It suffices to consider steps 2(a)ii, 2(a)iii, 2(b)ii, and 2c.

Consider the steps 2(a)ii and 2(a)iii. Let u' be a world of W' such that $r(u,u')$ holds. It is sufficient to show that $r_2(u_*,u')$ holds. Suppose that for every $1 \le i \le k$, $M, u \models B_i$ and if $B_i = \Box_t p$ then $Next(u, \Diamond_t\top)$ is defined. We need only to show that $M', u' \models A$. Since $r(u,u')$ holds, $M', u' \models (A \leftarrow B_1,\ldots,B_k)$. Hence, it is sufficient to show that $M', u' \models B_i$ for every $1 \le i \le k$. Fix such an index i. There are three cases to consider:

- Case $B_i = p$: Since $M, u \models B_i$, we have $p \in H(u)$. Since $r(u,u')$ holds, it follows that $M', u' \models B_i$.

- Case $B_i = \Diamond_t p$: Thus there exists $w \in W$ such that $R_t(u,w)$ holds and $p \in H(w)$. By Lemma 5, there exist $w_0,\ldots,w_k$ in W with $w_0 = u$, $w_k = w$, and indices $s_1,\ldots,s_k \in \mathcal{MOD}$ such that $R'_{s_i}(w_{i-1},w_i)$ holds for $1 \le i \le k$, and $R_{s_1} \circ \ldots \circ R_{s_k} \subseteq R_t$ follows from the L-frame restrictions, which we denote by (*). Let $\psi_1,\ldots,\psi_k$ be formulae of $\{\top\} \cup \mathcal{PROP}$ such that $Next(w_{i-1}, \Diamond_{s_i}\psi_i) = w_i$. By the definition of r and the inductive assumption, there exist $w'_0 = u', w'_1, \ldots, w'_k$ in W' such that $S_{s_i}(w'_{i-1}, w'_i)$ and $r(w_i, w'_i)$ hold for every $1 \le i \le k$. By (*), we have that $S_t(w'_0, w'_k)$ holds. Since $p \in H(w)$, $w = w_k$, and $r(w_k, w'_k)$ holds, we have $M', w'_k \models p$. It follows that $M', w'_0 \models \Diamond_t p$, which means $M', u' \models B_i$.

- Case $B_i = \Box_t p$: Let $w = Next(u, \Diamond_t\top)$. Since $M, u \models B_i$, we have that $p \in H(w)$. Let w' be an arbitrary world of W' such that $S_t(u',w')$ holds. By the inductive assumption, we have that $r(w,w')$. Hence $M', w' \models p$. It follows that $M', u' \models \Box_t p$, which means $M', u' \models B_i$.

Consider the step 2(b)ii. It suffices to show that if $r(u,u')$ holds, $Next(u, \Diamond_s \psi) = w$, $S_s(u',w')$ holds, and $M', w' \models \psi$ then $M', w' \models H(w_*)$. Suppose that $r(u,u')$ holds, $Next(u, \Diamond_s \psi) = w$, $S_s(u',w')$ holds, and $M', w' \models \psi$. By the inductive assumption, $M', w' \models H(w)$. Since $r(u,u')$ holds and $[A_t, Q]\psi \in H(u)$, it follows that $M', u' \models [A_t, Q]\psi$. Hence $M', w' \models [A_t, \delta_t(Q,s)]\psi$ (since $S_s(u',w')$ holds). Consequently, $M', w' \models H(w_*)$.

Consider the step 2c. Let w denote the world $\mathsf{Find}(\mathsf{CF}(\mathsf{Trans}(H(u),t)) \cup \{\psi\}, W, H)$. Suppose that $r(u,u')$ and $S_t(u',w')$ hold and $M', w' \models \psi$. It suffices to show that $M', w' \models H_2(w)$. Since $r(u,u')$ holds, $M', u' \models H(u)$. It follows that $M', w' \models \mathsf{Trans}(H(u),t)$ (since $S_t(u',w')$ holds). Hence $M', w' \models H_2(w)$. •

Lemma 8 *Let P be a positive logic program, L be a serial regular grammar logic, M be the model returned by Algorithm 1 for P and L, and $M' = \langle W', \tau', (S_t)_{t \in \mathcal{MOD}}, h' \rangle$ be an arbitrary L-model of P. Then $M \leq M'$.*

Proof. We will refer to the data structures used in Algorithm 1. Let r be the relation specified in Lemma 7 for the end of an execution of Algorithm 1 for P and L. By definition, $\forall\, x, x'\ r(x,x') \rightarrow h(x) \subseteq h(x')$ is true. By Lemma 7, $r(\tau, \tau')$ holds.

We first prove that $\forall\, t, x, x', y\ R_t(x,y) \wedge r(x,x') \rightarrow \exists y'\ S_t(x',y') \wedge r(y,y')$. Suppose that $R_t(x,y)$ and $r(x,x')$ hold. By Lemma 5, there exist $w_0, \ldots, w_k$ in W with $w_0 = x$, $w_k = y$, and indices $s_1, \ldots, s_k \in \mathcal{MOD}$ such that $R'_{s_i}(w_{i-1}, w_i)$ holds for $1 \leq i \leq k$, and $R_{s_1} \circ \ldots \circ R_{s_k} \subseteq R_t$ follows from the L-frame restrictions (denote this by (*)). Let $\psi_1, \ldots, \psi_k$ be formulae of $\{\top\} \cup \mathcal{PROP}$ such that $Next(w_{i-1}, \Diamond_{s_i} \psi_i) = w_i$. Let $w'_0 = x'$. Since $r(w_0, w'_0)$ holds and $M, w_0 \models \Diamond_{s_1} \psi_1$, we have that $M', w'_0 \models \Diamond_{s_1} \psi_1$. Let $w'_1 \in W'$ be the world such that $S_{s_1}(w'_0, w'_1)$ holds and $M', w'_1 \models \psi_1$. By Lemma 7, $r(w_1, w'_1)$ holds. Analogously, we claim that there exist $w'_0 = x', w'_1, \ldots, w'_k$ in W' such that $S_{s_i}(w'_{i-1}, w'_i)$ and $r(w_i, w'_i)$ hold for every $1 \leq i \leq k$. By (*), $S_t(w'_0, w'_k)$ holds. Hence, we can choose $y' = w'_k$.

We now prove that $\forall\, t, x, x', y'\ S_t(x',y') \wedge r(x,x') \rightarrow \exists y\ R_t(x,y) \wedge r(y,y')$. Suppose that $S_t(x',y')$ and $r(x,x')$ hold. Let $y = Next(x, \Diamond_t \top)$. Clearly, $R_t(x,y)$ holds. By Lemma 7, $r(y,y')$ also holds.

We have proved that $M \leq_r M'$. Therefore $M \leq M'$. •

Theorem 9 *Let P be a positive logic program and L be a serial regular grammar logic. Then the model M returned by Algorithm 1 for P and L is a least L-model of P.*

This theorem follows from Lemmas 6 and 8.

4 Lower Bounds

Let P denote a positive logic program and L denote a serial grammar logic. In this section, we give examples of P and L, for both of the case when L is fixed or P is fixed, such that every finite least L-model of P must have size $2^{\Omega(n)}$.

Lemma 10 *Let $M = \langle W, \tau, (R_t)_{t \in \mathcal{MOD}}, h \rangle$ be a finite Kripke model. Let Σ be a subset of $\mathcal{MOD}$ such that for every $t \in \Sigma$, R_t is a total function on W, i.e. $\forall x \exists! y. R_t(x,y)$. Let U be the subset of W defined by $w \in U$ iff there exists a path $\tau R_{t_1} w_1 R_{t_2} w_2 \ldots R_{t_k} w_k$ with $w_k = w$ and $t_1, \ldots, t_k \in \Sigma$. Suppose that $M' = \langle W', \tau', (R'_t)_{t \in \mathcal{MOD}}, h' \rangle$ is a finite Kripke model equivalent to M (i.e. $M' \equiv M$). Then for every $w \in U$ there exists $w' \in W'$ such that $(M,w) \equiv (M',w')$.*

Proof. It suffices to prove by induction on k that if $\tau R_{t_1} w_1 R_{t_2} w_2 \dots R_{t_k} w_k$ is a path in M with $t_1, \dots, t_k \in \Sigma$ then there exists $w'_k \in W'$ such that $(M, w_k) \equiv (M', w'_k)$, where $w_0 = \tau$.

The base case holds for $w'_0 = \tau'$. For the induction step, suppose that the hypothesis holds for k, and $R_{t_{k+1}}(w_k, w_{k+1})$ holds for some $t_{k+1} \in \Sigma$. We show that there exists $w'_{k+1} \in W'$ such that $(M, w_{k+1}) \equiv (M', w'_{k+1})$. Since $(M, w_k) \leq (M', w'_k)$ and $R_{t_{k+1}}(w_k, w_{k+1})$ holds, analogously as for the proof of Lemma 3, there exists $w'_{k+1} \in W'$ such that $R'_{t_{k+1}}(w'_k, w'_{k+1})$ holds and $(M, w_{k+1}) \leq (M', w'_{k+1})$. Once again, since $(M', w'_k) \leq (M, w_k)$ and $R'_{t_{k+1}}(w'_k, w'_{k+1})$ holds, analogously as for the proof of Lemma 3, there exists $w''_{k+1} \in W$ such that $R_{t_{k+1}}(w_k, w''_{k+1})$ holds and $(M', w'_{k+1}) \leq (M, w''_{k+1})$. Since $R_{t_{k+1}}$ is a function, $w''_{k+1} = w_{k+1}$, and hence $(M, w_{k+1}) \equiv (M', w'_{k+1})$. •

Proposition 11 *There are regular grammars G such that if L is a serial regular grammar logic corresponding to G then every least L-model of $\{\Box_s p\}$ has size $2^{\Omega(n)}$, where n is the size of G.*

Proof. Let $n > 0$ be a natural number. Consider the grammar G with rules $s \to a(x)^{n-1} \mid xs$ and $x \to a \mid b$. This grammar with s as the start symbol generates words over alphabet $\{a, b, x\}$ with a at the n-th last position. The automaton specifying this language is

$$A_s = \langle \{a, b, x, s\}, \{0, \dots, n\}, 0, \delta, \{n\} \rangle$$

where $\delta = \{(0, \sigma, 0), (k, \sigma, k+1) \mid \sigma \in \{a, b, x\} \text{ and } 1 \leq k < n\} \cup \{(0, a, 1)\}$.

Let L be the serial regular grammar logic corresponding to G and $M = \langle W, \tau, (R_t)_{t \in \mathcal{MOD}}, h \rangle$ be the least L-model of $\{\Box_s p\}$ constructed by Algorithm 1.

Let $\Sigma = \{a, b\}$. For $\alpha, \beta \in \Sigma^*$, define that $\alpha \sim \beta$ if for every $\gamma \in \Sigma^*$, $\alpha\gamma \in \mathcal{L}(A_s)$ iff $\beta\gamma \in \mathcal{L}(A_s)$. The equivalence relation $\sim$ has exactly 2^n abstract classes.

Let $\alpha, \beta \in \Sigma^*$ and $\alpha \nsim \beta$. Suppose that $\alpha = t_1 \dots t_k$ and $\beta = s_1 \dots s_h$. There exist w_α and w_β such that $(\tau, w_\alpha) \in R_{t_1} \circ \dots \circ R_{t_k}$ and $(\tau, w_\beta) \in R_{s_1} \circ \dots \circ R_{s_h}$. Since $\alpha \nsim \beta$, there exists $\gamma = \sigma_1 \dots \sigma_l \in \Sigma^*$ such that exactly one of $\alpha\gamma$ and $\beta\gamma$ belongs to $\mathcal{L}(A_s)$. Thus, $\Box_{\sigma_1} \dots \Box_{\sigma_l} p$ is true at exactly one of the worlds w_α and w_β. Hence $(M, w_\alpha) \equiv (M, w_\beta)$ does not hold. By Lemma 10, it follows that every least L-model of $\Box_s p$ has at least 2^n worlds. •

One can observe that the exponent $2^{\Omega(n)}$ in Proposition 11 can be completely explained from the determinization of the automaton A_s. That is, if A_s is a deterministic finite automaton, then there exists a least L-model of $\{\Box_s p\}$ that has a linear size. This is because of the chosen program $\{\Box_s p\}$. The observation is not necessarily true for other cases. For example, if the program is $\{\Box_s \Box_s p\}$ and the automaton A_s has several accepting states, then the situation is already more complicated. The following proposition states that in general the rank $2^{\Omega(n)}$ is not due to the chosen logic and its corresponding automata.

Proposition 12 *There exists a serial regular grammar logic L such that there are positive logic programs whose least L-models have size of rank $2^{\Omega(n)}$, where n is the size of the program.*

Proof. Consider the regular grammar consisting of the rules

$$\begin{aligned} 0 &\to a \mid b \\ 1 &\to a2 \\ 2 &\to \varepsilon \mid a2 \mid b2 \end{aligned}$$

Let L be the serial regular grammar logic corresponding to this grammar, where $\mathcal{MOD} = \{0, 1, 2, a, b\}$. Let $P = \{\Box_1 p_1,\ \Box_0 \Box_1 p_2,\ \Box_0 \Box_0 \Box_1 p_3,\ \dots,\ \Box_0^{n-1} \Box_1 p_n\}$ for some n and different

primitive propositions $p_1, \ldots, p_n$. Let $M = \langle W, \tau, (R_t)_{t \in \mathcal{MOD}}, h \rangle$ be the least L-model of P constructed by Algorithm 1. Let $\Sigma = \{a, b\}$ and U be the subset of W defined by $w \in U$ iff there exists a path $\tau R_{t_1} w_1 R_{t_2} w_2 \ldots R_{t_k} w_k$ with $w_k = w$ and $t_1, \ldots, t_k \in \Sigma$. Observe that for every subset X of $\{p_1, \ldots, p_n\}$, there exists $w \in U$ such that $h(w) = X$. Hence U contains at least 2^n worlds which are not equivalent to each other. By Lemma 10, it follows that every least L-model of P has at least 2^n worlds. •

5 Characterizing Serial Context-Free Grammar Logics Using Least *L*-Models

Theorem 13 *Let G be a context-free grammar and L be the serial grammar logic corresponding to G. Then there exists a finite least L-model of $\Box_s p$ iff the set of words derivable from s using G is a regular language.*

Proof. The "if" assertion follows from Algorithm 1 and Theorem 9. Consider the "only if" assertion. Suppose that $M = \langle W, \tau, (R_t)_{t \in \mathcal{MOD}}, h \rangle$ is a finite least L-model of $\Box_s p$. We show that the set of words derivable from s using G, which we denote by $\mathcal{L}(G, s)$, is a regular language.

Let $M' = \langle \mathcal{MOD}^*, \varepsilon, (R'_t)_{t \in \mathcal{MOD}}, h' \rangle$ be the Kripke model specified as follows: for $t \in \mathcal{MOD}$, R'_t is the least extension of $\{(u, ut) \mid u \in \mathcal{MOD}^*\}$ such that $(R'_t)_{t \in \mathcal{MOD}}$ satisfies all the L-frame restrictions; $h'(t_1 \ldots t_k) = \{p\}$ if $t_1 \ldots t_k$ is derivable from s by G, and $h'(t_1 \ldots t_k) = \emptyset$ otherwise.

We first study properties of M'.

Clearly, M' is an L-model of $\Box_s p$.

For $w \in \mathcal{MOD}^*$, let $\{w\}^{-1}\mathcal{L}(G, s)$ denote the set $\{w' \in \mathcal{MOD}^* \mid ww' \in \mathcal{L}(G, s)\}$. Let $r' \subseteq \mathcal{MOD}^* \times \mathcal{MOD}^*$ be $\{(w, w') \mid \{w\}^{-1}\mathcal{L}(G, s) \subseteq \{w'\}^{-1}\mathcal{L}(G, s)\}$. Observe that $h'(w) = \{p\}$ iff $w \in \mathcal{L}(G, s)$ iff $\varepsilon \in \{w\}^{-1}\mathcal{L}(G, s)$. It is easy to check that $M' \leq_{r'} M'$.

We show that for any $u, v \in \mathcal{MOD}^*$, if $R'_t(u, uv)$ holds than $(M', ut) \leq (M', uv)$, which means that ut is the least world among the worlds accessible from u via R'_t. Let $v = s_1 \ldots s_k$, where $s_1, \ldots, s_k \in \mathcal{MOD}$. Since $R'_t(u, uv)$ holds, we have that $R'_t \subseteq R'_{s_1} \circ \cdots \circ R'_{s_k}$ is a consequence of the L-frame restrictions, and hence $s_1 \ldots s_k$ is derivable from t using G (by Lemma 1). Hence $\{ut\}^{-1}\mathcal{L}(G, s) \subseteq \{uv\}^{-1}\mathcal{L}(G, s)$, and $r'(ut, uv)$ holds. Consequently, $(M', ut) \leq_{r'} (M', uv)$, and hence $(M', ut) \leq (M', uv)$.

As a consequence of the above assertion, for any $u, w \in \mathcal{MOD}^*$ and $t \in \mathcal{MOD}$, if $(M', u) \equiv (M', w)$ then $(M', ut) \equiv (M', wt)$. In fact, if φ is a positive formula such that $M', ut \models \varphi$ but $M', wt \not\models \varphi$ then $M', u \models \Box_t \varphi$ but $M', w \not\models \Box_t \varphi$.

We now study the relation between M' and M.

Let $r \subseteq \mathcal{MOD}^* \times W$ be defined as follows: $r(\varepsilon, \tau)$ holds; if there is a path $\tau R_{t_1} w_1 R_{t_2} w_2 \ldots w_{k-1} R_{t_k} w_k$ in M then $r(t_1 \ldots t_k, w_k)$ holds. It is easy to check that $M' \leq_r M$. Therefore M' is a least L-model of $\Box_s p$, and hence $M' \equiv M$.

We now define a function $f : \mathcal{MOD}^* \to W$ such that $(M', u) \equiv (M, f(u))$ and $R_t(f(u), f(ut))$ holds for every $u \in \mathcal{MOD}^*$ and $t \in \mathcal{MOD}$. Since $M' \equiv M$, let $f(\varepsilon) = \tau$. Assume that $f(u)$ is already defined, $(M', u) \equiv (M, f(u))$, and if $u = t_1 \ldots t_k$ then $R_{t_i}(f(t_1 \ldots t_{i-1}), f(t_1 \ldots t_i))$ holds for every $1 \leq i \leq k$. We recursively define $f(ut)$ as follows. Since $(M, f(u)) \leq (M', u)$, analogously as for the proof of Lemma 3, there exists $w \in W$ such that $R_t(f(u), w)$ holds and $(M, w) \leq (M', ut)$. Define $f(ut) = w$. Thus we have that $R_t(f(u), f(ut))$ holds and $(M, f(ut)) \leq (M', ut)$. By the definition of r, we also have that

$(M', ut) \leq (M, f(ut))$. Hence $(M', ut) \equiv (M, f(ut))$. Thus the function f satisfies the requirements. Furthermore, since $(M', ut) \equiv (M', wt)$ if $(M', u) \equiv (M', w)$, we can require also that, if $f(u) = f(w)$ then $f(ut) = f(wt)$ for any $u, w \in \mathcal{MOD}^*$ and $t \in \mathcal{MOD}$.

Define $S : W \times \mathcal{MOD} \to W$ to be the partial function such that $S(f(u), t) = f(ut)$ for $u \in \mathcal{MOD}^*$ and $t \in \mathcal{MOD}$. Define $S^* : W \times \mathcal{MOD}^* \to W$ to be the partial function such that $S^*(w, \varepsilon) = w$ and $S^*(f(u), vt) = S(S^*(f(u), v), t)$ for $u, v \in \mathcal{MOD}^*$ and $t \in \mathcal{MOD}$. Thus $f(uv) = S^*(f(u), v)$.

We now prove the claim of the theorem.

For $u, w \in \mathcal{MOD}^*$, define that $u \sim w$ if for every $v \in \mathcal{MOD}^*$, uv is derivable from s by G iff wv is derivable from s by G. By Myhill-Nerode theorem, the language consisting of words derivable from s using G is regular iff the equivalence relation $\sim$ has finitely many abstract classes. We show that if $f(u) = f(w)$ then $u \sim w$. Since the image of f is finite, this will imply that $\sim$ has finitely many abstract classes. Suppose that $f(u) = f(w)$. Let $v = t_1 \dots t_k$ be an arbitrary word over $\mathcal{MOD}$. The following conditions are equivalent

1. uv is derivable from s by G,
2. $h'(uv) = \{p\}$ (by the definition of M'),
3. $h(S^*(f(u), v)) = \{p\}$ (since $(M', uv) \equiv (M, f(uv))$ and $f(uv) = S^*(f(u), v)$)
4. $h(S^*(f(w), v)) = \{p\}$ (since $f(u) = f(w)$),
5. $h'(wv) = \{p\}$ (similarly as for item 3),
6. wv is derivable from s by G (by the definition of M').

Therefore $u \sim w$, which completes the proof. •

6 Conclusions

We have given an algorithm of constructing finite least Kripke models for positive logic programs in serial regular grammar logics. This class of logics is large and contains many useful multimodal logics (e.g., $KD4_{(m)}$, $S4_{(m)}$, the logics $KDI4$ and $KDI4_s$ for reasoning about multi-degree belief [22], the logic $KD4I_g$ for reasoning about beliefs of agents and groups of agents [24]). Our algorithm gives a bottom-up method for checking $P \models_L \varphi$ for a positive logic program P and a positive formula φ in a serial regular grammar logic L. The method is especially useful when P plays the role of a knowledge base that rarely changes, while φ is a query and varies.

The proof of correctness of our algorithm is simpler and shorter than our proof given for $KD4$ and $S4$ in [19] (3 pages of this paper in comparison with 11 pages in FI style). This is due to the use of a different technique for building model graphs. The algorithm given in this work uses graphs instead of trees and uses a special caching technique for building model graphs. These techniques are essential for getting the exponential upper bound for the time complexity and the size of the constructed model. When realizing a formula of the form $A \leftarrow B_1, \dots, B_k$ at a possible world w, we do not add A to the content of w, but just simulate the task by using another world. We also use such a technique for realizing formulae of the form $[A_t, Q]\psi$. Without this technique merging duplicates is necessary and the old nodes may need to be re-created later, and hence the performance is slowed down and complexity analysis could be difficult.

Our algorithm runs in exponential time and returns a model with exponential size. This exponential complexity is not surprising. However, we were able to prove a lower bound as well. We conjecture that the satisfiability problem of the Horn fragment of serial regular grammar logics is EXPTIME-hard (and therefore EXPTIME-complete). It remains an open problem. (The method of this work for checking L-satisfiability of $P \cup \{\neg\varphi\}$, where P is a positive logic program and φ is a positive formula, is to build a least L-model M of P and check whether $M \not\models \varphi$. Our conjecture is that: the complexity of the satisfiability problem is EXPTIME-hard, independently from the used method.)

Our method is adaptable for non-serial modal logics, in particular, for the deterministic Horn fragments of the description logic $\mathcal{ALC}$ [21] and test-free PDL [23]. When using a regular grammar logic as a description logic (where a primitive proposition stands for a concept, a modal operator stands for a role quantifier, and a positive logic program treated as global assumptions stands for a TBox), the seriality axioms do not hold anymore, and by putting a further restriction to obtain the deterministic Horn fragment as in [21, 23], one can show that the data complexity is reduced from coNP-complete to PTIME, which is interesting from a practical point of view.

Our results are interesting by themselves from a theoretical point of view. The theorem that "for G being a context-free grammar and L being the serial grammar logic corresponding to G, there exists a finite least L-model of $\Box_s p$ iff the set of words derivable from s using G is a regular language" is a nice theoretical result.

Acknowledgements

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

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. Baldoni, L. Giordano, and A. Martelli. A tableau for multimodal logics and some (un)decidability results. In *Proceedings of TABLEAUX'1998, LNCS 1397*, pages 44–59, 1998.

[4] M. Cadoli, L. Palopoli, and M. Lenzerini. Datalog and description logics: Expressive power. In S. Cluet and R. Hull, editors, *DBPL-6, LNCS 1369*, pages 281–298. Springer, 1998.

[5] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Data complexity of query answering in description logics. In I. Horrocks, U. Sattler, and F. Wolter, editors, *Description Logics 2005*.

[6] F. Debart, P. Enjalbert, and M. Lescot. Multimodal logic programming using equational and order-sorted logic. *Theoretical Comp. Science*, 105:141–166, 1992.

[7] S. Demri. The complexity of regularity in grammar logics and related modal logics. *Journal of Logic and Computation*, 11(6):933–960, 2001.

[8] S. Demri and H. de Nivelle. Deciding regular grammar logics with converse through first-order logic. *Journal of Logic, Language and Information*, 14(3):289–329, 2005.

[9] L. Fariñas del Cerro and M. Penttonen. Grammar logics. *Logique et Analyse*, 121-122:123–134, 1988.

[10] 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.

[11] B.N. Grosof, I. Horrocks, R. Volz, and S. Decker. Description logic programs: combining logic programs with description logic. In *WWW*, pages 48–57, 2003.

[12] D. Harel, D. Kozen, and J. Tiuryn. *Dynamic Logic*. MIT Press, 2000.

[13] I. Horrocks and U. Sattler. Decidability of SHIQ with complex role inclusion axioms. *Artificial Intelligence*, 160(1-2):79–104, 2004.

[14] U. Hustadt, B. Motik, and U. Sattler. Data complexity of reasoning in very expressive description logics. In L.P. Kaelbling and A. Saffiotti, editors, *Proceedings of IJCAI-05*, pages 466–471. Professional Book Center, 2005.

[15] K. Konolige. Belief and incompleteness. Technical Report 319, SRI Inter., 1984.

[16] A.Y. Levy and M.-Ch. Rousset. CARIN: A representation language combining Horn rules and description logics. In W. Wahlster, editor, *ECAI*, pages 323–327. John Wiley and Sons, Chichester, 1996.

[17] A. Mateescu and A. Salomaa. Formal languages: an introduction and a synopsis. In *Handbook of Formal Languages - Volume 1*, pages 1–40. Springer, 1997.

[18] J. McCarthy. First order theories of individual concepts and propositions. *Machine Intelligence*, 9:120–147, 1979.

[19] L.A. Nguyen. Constructing the least models for positive modal logic programs. *Fundamenta Informaticae*, 42(1):29–60, 2000.

[20] L.A. Nguyen. A fixpoint semantics and an SLD-resolution calculus for modal logic programs. *Fundamenta Informaticae*, 55(1):63–100, 2003.

[21] L.A. Nguyen. A bottom-up method for the deterministic Horn fragment of the description logic $\mathcal{ALC}$. In M. Fisher et al., editor, *Proceedings of JELIA 2006, LNAI 4160*, pages 346–358. Springer-Verlag, 2006.

[22] L.A. Nguyen. Multimodal logic programming. *Theoretical Computer Science*, 360:247–288, 2006.

[23] L.A. Nguyen. On the deterministic Horn fragment of test-free PDL. In I. Hodkinson and Y. Venema, editors, *Advances in Modal Logic - Volume 6*, pages 373–392. King's College Publications, 2006.

[24] L.A. Nguyen. Reasoning about epistemic states of agents by modal logic programming. In F. Toni and P. Torroni, editors, *Proceedings of CLIMA VI, LNAI 3900*, pages 37–56. Springer-Verlag, 2006.

[25] 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.

[26] H.J. Ohlbach. A resolution calculus for modal logics. In E.L. Lusk and R.A. Overbeek, editors, *Proceedings of CADE-88, LNCS 310*, pages 500–516. Springer, 1988.

[27] J. van Benthem. Correspondence theory. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic, Vol II*, pages 167–247. Reidel, Dordrecht, 1984.

[28] M. Wessel. Obstacles on the way to qualitative spatial reasoning with description logics: Some undecidability results. In *Description Logics 2001*.