Tableaux for Regular Grammar Logics of Agents Using Automaton-Modal Formulae

Rajeev Goré (rajeev.gore@anu.edu.au) *
The Australian National University and NICTA
Canberra ACT 0200, Australia

Linh Anh Nguyen (nguyen@mimuw.edu.pl)
Institute of Informatics, University of Warsaw
ul. Banacha 2, 02-097 Warsaw, Poland

Abstract. We present sound and complete tableau calculi for the class of regular grammar logics and a class eRG of extended regular grammar logics which contains useful epistemic logics for reasoning about beliefs of agents. Our tableau rules use a special feature called automaton-modal formulae which are similar to formulae of automaton propositional dynamic logic. Our calculi are cut-free and have the analytic superformula property so they give decision procedures. We show that the known EXPTIME upper bound for regular grammar logics can be obtained using our tableau calculus. We also prove that the general satisfiability problem of eRG logics is EXPTIME-complete.

Keywords: modal logics, regular grammar logics, tableaux, logics of agency.

1. Introduction

Multimodal (description) logics are useful in many areas of computer science: for example, multimodal logics are used in knowledge representation by interpreting $[i] \varphi$ as “agent $i$ knows/believes that $\varphi$ is true” [7, 14, 1]. Grammar logics are normal multimodal logics characterised by “inclusion” axioms like $[t_1] \ldots [t_h] \varphi \supset [s_1] \ldots [s_k] \varphi$, where $[t_i]$ and $[s_j]$ are modalities indexed by members $t_i$ and $s_j$ from some fixed set of indices. Thus $[1][2] \varphi \supset [1] \varphi$ captures “if agent one knows that agent two knows $\varphi$, then agent one knows $\varphi$”.

Inclusion axioms correspond in a strict sense to grammar rules of the form $t_1 t_2 \ldots t_h \rightarrow s_1 s_2 \ldots s_k$ when the index set is treated as a set of atomic words and juxtaposition is treated as word composition. Various refinements ask whether the corresponding grammar is left or right linear, or whether the language generated by the corresponding grammar is regular, context-free etc.

* National ICT Australia (NICTA) is funded through the Australian Government’s Backing Australia’s Ability initiative, in part through the Australian Research Council.

The satisfiability problem for logic $L$ is to check whether an input formula $\varphi$ is $L$-satisfiable. The general satisfiability problem for a class $C$ of logics is to check whether an input formula $\varphi$ is $L$-satisfiable in an input logic $L \in C$.

Grammar logics were introduced by Fariñas del Cerro and Penttonen in [8] and have been studied widely [3, 4, 21, 12, 5]. Baldoni et al. [3] gave a prefixed tableau calculus for grammar logics and used it to show that the general satisfiability problem of right linear grammar logics is decidable and the general satisfiability problem of context-free grammar logics is undecidable. But the techniques of Baldoni et al. cannot be easily extended to regular grammar logics.

While trying to understand why the decidability proof by Baldoni et al. [3, 2] cannot be naturally extended to left linear grammars, Demri [4] observed that although right linear grammars generate the same class of languages as left linear grammars, this correspondence is not useful at the level of regular grammar logics. By using a transformation into the satisfiability problem for propositional dynamic logic (PDL), Demri was able to prove that the general satisfiability problem of regular grammar logics is EXPTIME-complete and that the general satisfiability problem of linear grammar logics is undecidable. In [5], Demri and de Nivelle gave a translation of the satisfiability problem for grammar logics with converse into the two-variable guarded fragment $GF^2$ of first-order logic, and showed that the general satisfiability problem for regular grammar logics with converse is in EXPTIME. The relationship between grammar logics and description logics was considered, among others, in [12, 21].

Thus, various methods have been required to obtain complexity results and decision procedures for regular grammar logics. Efficient tableaux for propositional multimodal (description) logics are highly competitive with translation methods, so it is not at all obvious that the translation into $GF^2$ from [5] is the best method for deciding these logics. We therefore give a (non-prefixed) tableau calculus which is a decision procedure for the whole class of regular grammar logics, and which also gives an estimate of the complexity of these logics.

The naive way to encode inclusion axioms in a non-prefixed tableau calculus is to add a rule like $([t])$ shown below at left. But such rules cannot lead to a general decision procedure because there are well-known examples like transitivity $[t]\varphi \supset [t][t]\varphi$, whose analogous rule is shown below at right, which immediately cause an infinite branch by adding $[t][t]\varphi$, and then $[t][t][t]\varphi$, and so on:

\[
\frac{X; [t]\varphi}{X; [t][\varphi; [s_1][s_2] \ldots [s_k]\varphi}} \quad (\text{3f}) \quad \frac{X; [t]\varphi}{X; [t][\varphi; [t][t]\varphi}}
\]
Our calculus for regular grammar logics uses a special feature called automaton-modal formulae, which are similar to formulae of APDL [11]. Informally, whenever a formula $[t]\varphi$ is true at a tableau node $w$, we add an automaton-modal formula that tracks the modal transitions from $w$. If a sequence of transitions leads to a tableau node $u$, and this sequence corresponds to a word $s_1s_2\ldots s_k$ recognised by the automaton-modal formula, then we add the formula $\varphi$ to $u$. This captures the effect of the rule $([t])$ above left in a tractable manner since the influence of $[t]\varphi$ being true at $w$ can be computed directly from the content of the automaton-modal formulae in node $u$.

Although the class of regular grammar logics is large, it excludes the class of euclidean logics which have proved useful for reasoning about beliefs of agents. We therefore define a class $eRG$ of extended regular grammar logics, where each $eRG$ logic is a multimodal logic $L$ with: a base set of inclusion axioms corresponding to a regular grammar; a set of “serial” modal indices with corresponding axioms $D$; and a set of “weak-S5 grammar terminal” modal indices with corresponding axioms $D$, 4, and 5. The class of $eRG$ logics is important as it contains useful epistemic logics for reasoning about beliefs of agents.

We then extend our conference paper [10] by giving a tableau calculus for the class $eRG$ of extended regular grammar logics. Our tableau calculi are sound, complete, cut-free and have the analytic superformula property, so they are decision procedures. As usual for tableau calculi, they allow efficient implementation and good complexity estimation. We omit the details of proving interpolation, which can be found in [10].

The rest of this paper is structured as follows. In Section 2, we define regular grammar logics and automaton-modal formulae. In Section 3, we present our tableau calculus for regular grammar logics, and prove it sound and complete. In Section 4, we introduce the class of $eRG$ logics. In Section 5, we present our tableau calculus for $eRG$ logics, and prove it sound and complete. In Section 6, we prove that the general satisfiability problem of regular grammar logics and $eRG$ logics is in EXPTIME by using our tableau rules in a systematic way. Further work and concluding remarks are in Section 7.

2. Preliminaries

2.1. Definitions for Multimodal Logics

Our modal language is built from two disjoint sets: $MOD$ is a finite set of modal indices and $PROP$ is a set of primitive propositions. We use $p$ and $q$ for elements of $PROP$ and use $t$ and $s$ for elements of $MOD$. 


Formulae of our primitive language are recursively defined using the BNF grammar below:

\[ \phi ::= p \mid \neg \phi \mid \phi \land \phi \mid \phi \lor \phi \mid \phi \supset \phi \mid [t] \phi \mid (t) \phi. \]

A Kripke frame is a tuple \( \langle W, \tau, \{ R_t \mid t \in \text{MOD} \} \rangle \), where \( W \) is a nonempty set of possible worlds, \( \tau \in W \) is the current world, and each \( R_t \) is a binary relation on \( W \), called the accessibility relation for \( [t] \) and \( (t) \). If \( R_t(w, u) \) holds then we say that the world \( w \) sees world \( u \) via \( R_t \).

A Kripke model is a tuple \( \langle W, \tau, \{ R_t \mid t \in \text{MOD} \}, h \rangle \), where \( \langle W, \tau, \{ R_t \mid t \in \text{MOD} \} \rangle \) 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 \( \langle W, \tau, \{ R_t \mid t \in \text{MOD} \}, H \rangle \), where \( \langle W, \tau, \{ R_t \mid t \in \text{MOD} \} \rangle \) is a Kripke frame and \( H \) is a function mapping worlds to formula sets. We sometimes treat model graphs as models with the range of \( H \) restricted to \( \text{PROP} \).

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

- \( M, w |\mids [t] \phi \iff \forall v \in W. R_t(w, v) \) implies \( M, v |\mids \phi \)
- \( M, w |\mids (t) \phi \iff \exists v \in W. R_t(w, v) \) and \( M, v |\mids \phi \).

We say that \( \phi \) is satisfied at \( w \) in \( M \) if \( M, w |\mids \phi \). We say that \( \phi \) is satisfied in \( M \) and call \( M \) a model of \( \phi \) if \( M, \tau |\mids \phi \).

If we consider only Kripke models, with no restrictions on \( R_t \), we obtain a normal multimodal logic with a standard Hilbert-style axiomatisation \( K_n \).

Note: We now assume that formulae are in negation normal form, where \( \supset \) is translated away and \( \neg \) occurs only directly before primitive propositions. It is well-known that every formula \( \phi \) has a logically equivalent formula \( \phi' \) which is in negation normal form.

2.2. Regular Grammar Logics

Recall that a finite automaton \( A \) is a tuple \( \langle \Sigma, Q, I, \delta, F \rangle \), where: \( \Sigma \) is the alphabet (for our case \( \Sigma = \text{MOD} \)); \( Q \) 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 \( s_1 \ldots s_k \) is a finite sequence of states \( q_0, q_1, \ldots, q_k \) 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/recognised by \( A \) is \( \mathcal{L}(A) \).
Given two binary relations $R_1$ and $R_2$ over $W$, their relational composition $R_1 \circ R_2 = \{(x, y) \mid \exists y \in W. R_1(x, y) \& R_2(y, z)\}$ is also a binary relation over $W$.

A grammar logic is a multimodal logic extending $K_n$ with “inclusion axioms” of the form $[t_1] \ldots [t_h] \varphi \supset [s_1] \ldots [s_k] \varphi$, where $\{t_1, \ldots, t_h, s_1, \ldots, s_k\} \subseteq \mathcal{MOD}$. Each inclusion axiom corresponds to the restriction $R_{s_1} \circ \ldots \circ R_{s_k} \subseteq R_{t_1} \circ \ldots \circ R_{t_h}$ on accessibility relations where the corresponding side stands for the identity relation if $k = 0$ or $h = 0$. For a grammar logic $L$, the $L$-frame restrictions are the set of all such corresponding restrictions. A Kripke model is an $L$-model if its frame satisfies all $L$-frame restrictions. A formula $\varphi$ is $L$-satisfiable if there exists an $L$-model satisfying it. A formula $\varphi$ is $L$-valid if it is satisfied in all $L$-models.

An inclusion axiom $[t_1] \ldots [t_h] \varphi \supset [s_1] \ldots [s_k] \varphi$ can be seen as the grammar rule $t_1 \ldots t_k \to s_1 \ldots s_k$ where the corresponding side stands for the empty word if $k = 0$ or $h = 0$. Thus the inclusion axioms of a grammar logic $L$ capture a grammar $G(L)$ where we do not distinguish between terminal symbols and nonterminal symbols. The grammar $G(L)$ is context-free if its rules are of the form $t \to s_1 \ldots s_k$, and is regular if it is context-free and for every $t \in \mathcal{MOD}$ there exists a finite automaton $A_t$ that recognises the words derivable from $t$ using $G(L)$.

A regular grammar logic $L$ is a grammar logic whose inclusion axioms correspond to grammar rules that collectively capture a regular grammar $G(L)$. 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 there is no syntactic way to specify the class of regular (context-free) grammars, and checking whether a context-free grammar generates a regular language is undecidable (see, e.g., [13]). Hence, we cannot compute these automata from the axioms of an arbitrary regular grammar logic in general. We therefore assume that for each $t \in \mathcal{MOD}$ we are given an automaton $A_t$ recognising the words derivable from $t$ using $G(L)$. These are the automata specifying $L$.

**Lemma 1.** Let $L$ be a regular grammar logic and let $\{A_t \mid t \in \mathcal{MOD}\}$ be the automata specifying $L$. Then the following conditions are equivalent:

(i) the word $s_1 \ldots s_k$ is accepted by $A_t$

(ii) the formula $[t] \varphi \supset [s_1] \ldots [s_k] \varphi$ is $L$-valid

(iii) the inclusion $R_{s_1} \circ \ldots \circ R_{s_k} \subseteq R_t$ is a consequence of the $L$-frame restrictions.
Proof. The equivalence \((ii) \iff (iii)\) is well-known from correspondence theory \([20]\). The implication \((i) \Rightarrow (ii)\) follows by induction on the length of the derivation of \(s_1 \ldots s_k\) from \(t\) by the grammar \(G(L)\), using substitution, the K-axiom \([t](\varphi \supset \psi) \supset ([t] \varphi \supset [t] \psi)\) and the modal necessitation rule \(\varphi/\psi\). The implication \((iii) \Rightarrow (i)\) 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. See also \([3, 4]\) for details.

2.3. Automaton-Modal Formulae

If \(A\) is a finite automaton, \(Q\) is a subset of the states of \(A\), and \(\varphi\) 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 written as \((A,Q) : \varphi\) and called an automaton-labelled formula.

Fix a serial regular grammar logic \(L\) and let \(\{A_t = (M\text{O}D, Q_t, I_t, \delta_t, F_t) \mid t \in M\text{O}D\}\) be the automata specifying \(L\). Let \(\varepsilon\) be the empty word, let \(s \in M\text{O}D\), let \(\alpha\) be a (possibly empty) word and define:

\[
\tilde{\delta}_t(Q, \varepsilon) = Q \quad \tilde{\delta}_t(Q, \alpha s) = \delta_t(\tilde{\delta}_t(Q, \alpha), s).
\]

The semantics of formulae with automaton-modal operators are defined as below:

\[
M, w_0 \models [A_t, Q] \varphi \text{ if } M, w_k \models \varphi \text{ for every path } w_0 R_{s_1} w_1 \ldots R_{s_k} w_k
\]

with \(k \geq 0\) and \(\tilde{\delta}_t(Q, s_1 \ldots s_k) \cap F_t \neq \emptyset\).

That is, \(s_1 \ldots s_k\) is accepted by \(A_t\) when starting from some state from \(Q\).

The intuitions of formulae with automaton-modal operators are as follows:
A formula of the form $\lozenge t \varphi$ 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 $\varphi$ to $u$.

Our formulae with automaton-modal operators are similar to formulae of automaton propositional dynamic logic (APDL) [1]. A formula involving automata in APDL is of the form $[A] \varphi$, where $A$ is a finite automata with one initial state and one accepting state. A formula like our $[A_t, Q] \varphi$ with $Q = \{q_1, q_2, ..., q_k\}$ can be simulated by the APDL formula $[B_1] \varphi \lor ... \lor [B_k] \varphi$ 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. Our compact representation not only saves memory but also increases efficiency of deduction.

From now on, by a formula we mean either a formula in the primitive language (as defined in Section 2.1) or an automaton-modal formula.

### 2.4. Definitions for Tableau Calculi

As in our previous works on tableau calculi [9, 16], our tableaux trace their roots to Hintikka via [19]. A tableau rule $\sigma$ consists of a numerator $N$ above the line and a (finite) list of denominators $D_1, D_2, ..., D_k$ (below the line) separated by vertical bars. The numerator is a finite formula set, and so is each denominator. As we shall see later, each rule is read downwards as “if the numerator is $L$-satisfiable, then so is one of the denominators”. The numerator of each tableau rule contains one or more distinguished formulae called the principal formulae. A tableau calculus $\mathcal{CL}$ for a logic $L$ is a finite set of tableau rules.

A $\mathcal{CL}$-tableau for a finite set $X$ of formulae is a tree with root $X$ whose nodes carry finite formula sets obtained from their parent nodes by instantiating a tableau rule with the proviso that if a child $s$ carries a set $Z$ and $Z$ has already appeared on the branch from the root to $s$ then $s$ is an end node.

Let $\Delta$ be a set of tableau rules. We say that $Y$ is obtainable from $X$ by applications of rules from $\Delta$ if there exists a tableau for $X$ which uses only rules from $\Delta$ and has a node that carries $Y$. A node to which
Table I. Tableau Rules

\[
\begin{align*}
(\bot) & \quad X; p; \neg p \\
(\land) & \quad X; \varphi \land \psi \\
(\lor) & \quad X; \varphi \lor \psi \\
(label) & \quad X; [A_t, I_t] \varphi \\
(add) & \quad X; [A_t, Q] \varphi \\
\text{(trans)} & \quad X; [\langle t \rangle, \varphi] \\
\end{align*}
\]

no rule is applicable is also an end-node. A branch in a tableau is closed if its end node carries only \( \bot \). A tableau is closed if every one of its branches is closed. A tableau is open if it is not closed. A finite formula set \( X \) in the primitive language is CL-consistent if every CL-tableau for \( X \) is open. If there is a closed CL-tableau for \( X \) then \( X \) is CL-inconsistent.

A tableau calculus CL is sound if for all finite formula sets \( X \) in the primitive language, \( X \) is L-satisfiable implies \( X \) is CL-consistent. It is complete if for all finite formula sets \( X \) in the primitive language, \( X \) is CL-consistent implies \( X \) is L-satisfiable. Let \( \sigma \) be a rule of CL. We say that \( \sigma \) is sound w.r.t. L if for every instance \( \sigma' \) of \( \sigma \), if the numerator of \( \sigma' \) is L-satisfiable then so is one of the denominators of \( \sigma' \). Any calculus CL containing only rules sound w.r.t. L is sound.

3. A Tableau Calculus for Regular Grammar Logics

Fix a regular grammar logic L and let \( \{ A_t = \langle \text{MOD}, Q_t, I_t, \delta_t, F_t \rangle \mid t \in \text{MOD} \} \) be the automata specifying L. Recall that formulae are in negation normal form. We use \( X \) for a formula set. The transfer of \( X \) through \( \langle t \rangle \), denoted by \( \text{trans}(X, t) \), is:

\[
\text{trans}(X, t) = \{ [A_s, \delta_s(Q, t)] \psi \mid [A_s, Q] \psi \in X \}.
\]

The tableau calculus CL is given in Table I. The first six rules are static rules, and the last rule (trans) is a transitional rule.

Example 2. Let \( \text{MOD} = \{0, 1, 2\} \). Consider the grammar logic L with the inclusion axioms \( [0] \varphi \supset \varphi \) and \([i] \varphi \supset [j][k] \varphi \) if \( i = (j+k) \mod 3 \). This is a regular grammar logic because the corresponding grammar is regular. We have \( A_i = \langle \text{MOD}, \text{MOD}, \{0\}, \delta, \{i\} \rangle \) for \( i \in \text{MOD} \), where \( \delta = \{(j, k, l) \mid j, k, l \in \{0, 1, 2\} \text{ and } l = (j+k) \mod 3 \} \).
In Figure 1, we give a closed $CL$-tableau for $X = \{ (0)p, [0](\neg p \lor (1)q), [1](\neg q \lor (2)r), [0]\neg r \}$, in which principal formulae of nodes are underlined. By the soundness lemma proved shortly, $X$ is $L$-unsatisfiable. The arrows stand for rule applications and are annotated with the rule name. The labels $R_i$ for $i \in \{ 0, 1, 2 \}$ to the right of the arrows marked with (trans)-rule applications stand for the label on the associated edges in the underlying model being explored by the tableau.
A tableau calculus $\mathcal{C}L$ has the analytic superformula property iff to every finite set $X$ we can assign a finite set $X_{\mathcal{C}L}$ which contains all formulae that may appear in any tableau for $X$. We write $Sf(\varphi)$ for the set of all subformulae of $\varphi$, and $Sf(X)$ for the set $\bigcup_{\varphi \in X} Sf(\varphi) \cup \{\bot\}$. Our calculus has the analytic superformula property, with

$$X_{\mathcal{C}L} = Sf(X) \cup \{[A_t, Q]\varphi \mid [t]\varphi \in Sf(X) \& Q \subseteq Q_t\}.$$ 

3.1. Soundness

**Lemma 2.** The tableau calculus $\mathcal{C}L$ is sound.

*Proof.* We show that $\mathcal{C}L$ contains only rules sound w.r.t. $L$ as follows. Suppose that the numerator of the considered rule is satisfied at a world $w$ in an $L$-model $M = \langle W, \tau, \{R_t \mid t \in \text{MOD}\}, h \rangle$. We have to show that at least one of the denominators of the rule is also $L$-satisfiable.

For the static rules, we show that some denominator is satisfied at $w$ itself. For the transitional rule (trans), we show that its denominator is satisfied at some world reachable from $w$ via $R_t$ in the same $L$-model.

($\bot$), ($\land$), ($\lor$): These cases are obvious.

(label): Suppose that $M, w \models X; [t]\varphi$. Let $w_0 = w, w_1, \ldots, w_k$ be worlds of $M$ such that $R_{s_i}(w_{i-1}, w_i)$ holds for $1 \leq i \leq k$ and $s_1 \ldots s_k$ is accepted by $A_t$. By Lemma 1, $[t]\psi \supset [s_1] \ldots [s_k]\psi$ is $L$-valid. Hence $M, w_k \models \varphi$. Thus, $M, w \models [A_t, I_t]\varphi$.

($\cup$), (add): These cases follow from the semantics of automaton-modal operators.

(trans): Suppose that $M, w \models X; (t)\varphi$. Then there exists some $u$ such that $R_t(w, u)$ holds and $M, u \models \varphi$. For each $[A_s, Q]\psi \in X$, we have $M, w \models [A_s, Q]\psi$, and by the semantics of automaton-modal formulae, it follows that $M, u \models [A_s, \delta_s(Q, t)]\psi$. Hence, the denominator is satisfied at $u$.

3.2. Proving Completeness via Model Graphs

We prove completeness of our calculus via model graphs following [19, 9, 15, 16] by giving an algorithm that accepts a finite $\mathcal{CL}$-consistent formula set $X$ in the primitive language and constructs an $L$-model graph (defined below) for $X$ that satisfies every one of its formulae at the appropriate world.
In the rules $(\land), (\lor), (\text{label}), (\cup)$ the principal formulae do not occur in the denominators. For any of these rules $\rho$, let $\rho'$ denote the rule obtained from $\rho$ by adding the principal formula to each of the denominators. Let $\mathcal{SCL}$ denote the set of static rules of $\mathcal{CL}$ with $(\land), (\lor), (\text{label}), (\cup)$ replaced by $(\land'), (\lor'), (\text{label}'), (\cup')$. For every rule of $\mathcal{SCL}$, except $(\bot)$, the numerator is included in each of the denominators. It is obvious that the new rules are still sound.

For a finite $\mathcal{CL}$-consistent formula set $X$, a formula set $Y$ is called a $\mathcal{CL}$-saturation of $X$ if $Y$ is a maximal $\mathcal{CL}$-consistent set obtainable from $X$ by applications of the rules of $\mathcal{SCL}$. A set $X$ is closed w.r.t. a tableau rule if applying that rule to $X$ gives back $X$ as one of the denominators.

**Lemma 3.** Let $X$ be a finite $\mathcal{CL}$-consistent formula set and $Y$ a $\mathcal{CL}$-saturation of $X$. Then $X \subseteq Y \subseteq X^*_\mathcal{CL}$ and $Y$ is closed w.r.t. the rules of $\mathcal{SCL}$. Furthermore, there is an effective procedure that, given a finite $\mathcal{CL}$-consistent formula set $X$, constructs some $\mathcal{CL}$-saturation of $X$.

**Proof.** It is clear that $X \subseteq Y \subseteq X^*_\mathcal{CL}$. Observe that if a rule of $\mathcal{SCL}$ is applicable to $Y$, then one of the corresponding instances of the denominators is $\mathcal{CL}$-consistent. Since $Y$ is a $\mathcal{CL}$-saturation, $Y$ is closed w.r.t. the rules of $\mathcal{SCL}$.

We construct a $\mathcal{CL}$-saturation of $X$ as follows: let $Y = X$; while some static rule of $\mathcal{SCL}$ is applicable to $Y$ and has a corresponding denominator instance $Z$ which is $\mathcal{CL}$-consistent and strictly contains $Y$, set $Y = Z$. At each iteration, $Y \subset Z \subseteq X^*_\mathcal{CL}$, so this process always terminates. Clearly, the resulting set $Y$ is a $\mathcal{CL}$-saturation of $X$. \hfill $\Box$

A model graph is an $L$-model graph if its frame is an $L$-frame. An $L$-model graph $(W, \tau, \{R_t \mid t \in \mathcal{MOD}\}, H)$ is saturated if every $w \in W$ satisfies:

- if $\varphi \land \psi \in H(w)$ then $\{\varphi, \psi\} \subseteq H(w)$;
- if $\varphi \lor \psi \in H(w)$ then $\varphi \in H(w)$ or $\psi \in H(w)$;
- if $(t)^* \varphi \in H(w)$ and $R_t(w,u)$ holds then $\varphi \in H(u)$;
- if $(t) \varphi \in H(w)$ then $\exists u \in W$ with $R_t(w,u)$ and $\varphi \in H(u)$.

A saturated model graph is consistent if no world contains $\bot$, and no world contains $\{p, \neg p\}$. Our model graphs are merely a data structure, while Rautenberg’s are required to be saturated and consistent.

**Lemma 4.** If $M = (W, \tau, \{R_t \mid t \in \mathcal{MOD}\}, H)$ is a consistent saturated $L$-model graph, then $M$ satisfies all formulae of $H(\tau)$ which are in the primitive language.
Proof. By proving \( \varphi \in H(w) \) implies \( M, w \models \varphi \) via induction on the length of \( \varphi \).

Given a finite \( CL \)-consistent set \( X \) in the primitive language, we construct a consistent saturated \( L \)-model graph \( M = \langle W, \tau, \{ R_t \mid t \in \text{MOD} \}, H \rangle \) such that \( X \subseteq H(\tau) \), thereby giving an \( L \)-model for \( X \).

3.3. Constructing Model Graphs

Given \( X \), the compact form \( \text{compact}(X) \) of \( X \) is the least set such that:

- if \( \varphi \in X \) and \( \varphi \) is not of the form \( [A_t, Q]\psi \) then \( \varphi \in \text{compact}(X) \);
- if \( [A_t, Q_1, \ldots, Q_k]\psi \in X \) 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 \( [A_t, Q_1 \cup \ldots \cup Q_k]\psi \in \text{compact}(X) \).

The compact form of \( X \) is obtainable from \( X \) by applications of rule \( (\cup) \).

In the following algorithm, the worlds of the constructed model graph are marked either as \textit{unresolved} or as \textit{resolved}.

**ALGORITHM 1.**
Input: a finite \( CL \)-consistent set \( X \) of primitive language formulae.
Output: an \( L \)-model graph \( M = \langle W, \tau, \{ R_t \mid t \in \text{MOD} \}, H \rangle \) of \( X \).

1. Let \( W = \{ \tau \} \) and \( R_t' = \emptyset \) for all \( t \in \text{MOD} \).
   Let \( Y \) be a \( CL \)-saturation of \( X \) and let \( H(\tau) = \text{compact}(Y) \).
   Mark \( \tau \) as unresolved.

2. While there are unresolved worlds, take one, say \( w \), and do:
   a) For every formula \( \langle t \rangle \varphi \) in \( H(w) \):
      i) Let \( U = \text{trans}(H(w), t) \cup \{ \varphi \} \) be the result of applying rule \( (\text{trans}) \) to \( H(w) \), let \( Y \) be a \( CL \)-saturation of \( U \), and let \( Z = \text{compact}(Y) \).
      ii) If \( \exists u \in W \) on the path from the root to \( w \) with \( H(u) = Z \), then add the pair \( (w, u) \) to \( R_t' \). Otherwise, add a new world \( u \) with content \( Z \) to \( W \), mark it as unresolved, and add the pair \( (w, u) \) to \( R_t' \).
   b) Mark \( w \) as resolved.

3. Let \( R_t \) be the least extension of \( R_t' \) for \( t \in \text{MOD} \) such that \( \langle W, \tau, \{ R_t \mid t \in \text{MOD} \} \rangle \) is an \( L \)-frame.
This algorithm always terminates: eventually, for every \( w \), either \( w \) contains no \((t)\)-formulae, or there exists an ancestor with \( H(w) = Z \) at Step 2(a)ii because all \( CL \)-saturated sets are drawn from the finite and fixed set \( X^*_{CL} \).

**LEMMA 5.** Suppose \( R_t(w,u) \) holds via Step 3. Then there exist \( w_0, \ldots, w_k \) in \( M \) with \( w_0 = w, w_k = u \), and indices \( s_1, \ldots, s_k \in \text{MOD} \) such that \( R'_{s_i}(w_{i-1}, w_i) \) holds for \( 1 \leq i \leq k \), and \( R_{s_1} \circ \cdots \circ R_{s_k} \subseteq R_t \) follows from the \( L \)-frame restrictions.

**Proof.** By induction on number of inferences in deriving \( R_t(w,u) \) when extending \( R'_{s_i} \) to \( R_s \) for \( s \in \text{MOD} \), with \( L \)-frame restrictions of the form \( R_{t_1} \circ \cdots \circ R_{t_k} \subseteq R_s \). \( \Box \)

### 3.4. Completeness Proof

**LEMMA 6.** Let \( X \) be a finite \( CL \)-consistent set of formulae in the primitive language and \( M = \langle W, \tau, \{ R_t \mid t \in \text{MOD} \}; H \rangle \) be the model graph for \( X \) constructed by Algorithm 1. Then \( M \) is a consistent saturated \( L \)-model graph satisfying \( X \).

**Proof.** It is clear that \( M \) is an \( L \)-model graph and for any \( w \in W \), the set \( H(w) \) is \( CL \)-consistent. We want to show that \( M \) is a saturated model graph. It suffices to show that, for every \( w, u \in W \), if \( [t] \varphi \in H(w) \) and \( R_t(w,u) \) holds then \( \varphi \in H(u) \). Suppose that \( [t] \varphi \in H(w) \) and \( R_t(w,u) \) holds. By Lemma 5, there exist worlds \( w_0, \ldots, w_k \) with \( w_0 = w, w_k = u \) and indices \( s_1, \ldots, s_k \in \text{MOD} \) such that \( R'_{s_i}(w_{i-1}, w_i) \) holds for \( 1 \leq i \leq k \), and \( R_{s_1} \circ \cdots \circ R_{s_k} \subseteq R_t \) is a consequence of the \( L \)-frame restrictions. Since \( [t] \varphi \in H(w) \), there exists \( [A_t, Q] \varphi \in H(w) \) such that \( Q \supseteq I_t \) by rule (label'). By Step 2a of Algorithm 1, there exists \( [A_t, Q_i] \varphi \in H(w_i) \) such that \( Q_i \supseteq \delta_i(I_t, s_1 \ldots s_i) \) for \( 1 \leq i \leq k \). Hence \( Q_k \supseteq \delta_k(I_t, s_1 \ldots s_k) \) and \( [A_t, Q_k] \varphi \in H(u) \). Since \( R_{s_1} \circ \cdots \circ R_{s_k} \subseteq R_t \) is a consequence of the \( L \)-frame restrictions, by Lemma 1, the word \( s_1 \ldots s_k \) is accepted by \( A_t \). Hence \( \delta_k(I_t, s_1 \ldots s_k) \cap F_t \neq \emptyset \) and \( Q_k \cap F_t \neq \emptyset \). It follows that \( \varphi \in H(u) \), since \( H(u) \) is the compact form of a \( CL \)-saturation. \( \Box \)

The following theorem follows from Lemmas 2 and 6.

**THEOREM 7.** The tableau calculus \( CL \) is sound and complete.

### 4. A Class \( eRG \) of Extended Regular Grammar Logics

An extended regular grammar \( (eRG) \) logic is a multimodal logic \( L \) with:
Inclusion Axioms: a set \(\text{IA}(L)\) of inclusion axioms \([t]\phi \supset [s_1] \ldots [s_k]\phi\) with \(k \geq 0\) whose corresponding grammar rules \(t \rightarrow s_1 \ldots s_k\) jointly form a regular grammar \(RG(L)\); 

Seriality Axioms: a set \(\text{DI}(L) \subseteq \text{MOD}\) of D-indices with corresponding seriality axioms \([t]\phi \supset \langle t \rangle \phi\) for every \(t \in \text{DI}(L)\); 

Terminal 45-Condition: a set \(\text{EI}(L) \subseteq \text{DI}(L)\) of E-indices with corresponding axioms \([t]\phi \supset [t][t]\phi\) and \(\langle t \rangle \phi \supset [t][t]\phi\) for every \(t \in \text{EI}(L)\) and the condition that \(\text{IA}(L)\) contains no inclusion axioms of the form \([t]\phi \supset [s_1] \ldots [s_k]\phi\) for \(t \in \text{EI}(L)\).

Observe that for every \(eRG\) logic \(L\), every D-index \(t \in \text{DI}(L)\) is characterised by a serial accessibility relation \(R_t\), and every E-index \(t \in \text{EI}(L)\) is characterised by the frame restrictions of \(KD45\).

4.1. Usefulness of \(eRG\) Logics

We show that the class of \(eRG\) logics contains a particularly useful class of epistemic logics for reasoning about beliefs of agents. More specifically, modal logics of belief invariably utilise the following:

Belief Consistency: Since \(\langle t \rangle \phi \equiv \neg [t] \neg \phi\), the D-axiom \([t]\phi \supset \langle t \rangle \phi\) states that agents cannot believe both \(\phi\) and \(\neg \phi\).

Positive Introspection: The 4-axiom \([t]\phi \supset [t][t]\phi\) states that agents are aware of what they believe.

Negative Introspection: The 5-axiom \(\langle t \rangle \phi \supset [t]\langle t \rangle \phi\), or alternatively \(\neg [t] \psi \supset [t] \neg [t] \psi\), states that agents are aware of what they do not believe.

We do not require axiom \(D\) for every modal index in \(eRG\) logics, but we allow axiom 5 only for modal indices which satisfy the terminal 45-condition. This restriction is primarily motivated by technical reasons since the resulting logics are then amenable to our general (non-prefixed) tableau method. But this restriction can also be justified from practical considerations as shown next.

In [18], Nguyen studied a multimodal logic \(KD4I_s5a\) for reasoning about belief and common belief of agents in multi-agent systems. He adopted axioms \(D\) and 4 for all agents and groups of agents, and axiom \(I : [t]\phi \supset [s]\phi\) for any (proper) super-group \(t\) of agent (group) \(s\), but adopted axiom 5 \(\langle s \rangle \phi \supset [s]\langle s \rangle \phi\) only for single agents \(s\). If \(t\) is a non-singleton group and \(s\) is a single agent belonging to \(t\), then the contra-positive of \([t]\phi \supset [s]\phi\) gives us \(\langle s \rangle \phi \supset \langle t \rangle \phi\). If axiom 5 were present for the proper group \(t\) then \(\langle s \rangle \phi\) would give us \([t]\langle t \rangle \phi\). But
\(\langle s \rangle \varphi \supset [t][t] \varphi\) states that the belief of a single agent \(s\) leads to a belief among the whole super-group \(t\) about \(\varphi\). Conversely, the contra-positive \((t)[t] \varphi \supset [s] \varphi\) states that if the group jointly does not believe that it does not jointly believe \(\varphi\), then single agent \(s\) believes \(\varphi\), which seems equally absurd. Our restriction adopting axiom 5 only for modal indices which satisfy the terminal 45-condition is analogous to adopting axiom 5 only for single agents in the multimodal logic \(KDAI_g \delta_n\).

It can be shown that the multimodal logics of belief \(KDAI_4, KDAI_4s, KDAI_g, KDAI_g \delta_n\) studied by Nguyen in [17, 18], as well as \(KD45(m)\), belong to the class \(eRG\).

4.2. Some Properties of \(eRG\) Logics

Let \(L\) be an \(eRG\) logic. For \(t \in EI(L)\), logic \(L\) contains the axiom \((t) \varphi \supset [t][t] \varphi\), which can also be written as \((t)[t] \psi \supset [t] \psi\). This latter axiom is stronger than the inclusion axiom \([t][t] \psi \supset [t] \psi\) (because \(t \in EI(L) \subseteq DI(L)\) and \([t][t] \psi \supset (t)[t] \psi\) is \(L\)-valid), which corresponds to the grammar rule \(tt \rightarrow t\). Let \(eRG(L)\) be the grammar extending \(RG(L)\) with rules \(tt \rightarrow t\) for \(t \in EI(L)\). We call \(eRG(L)\) the extended grammar of \(L\). Syntactically, \(eRG(L)\) is not a regular grammar.

Let \(L\) be an \(eRG\) logic and let \(\{A_t \mid t \in MOD \setminus EI(L)\}\) be the automata specifying the regular grammar \(RG(L)\). An \(s\)-path from state \(q_0\) to state \(q_n\) in \(A_t\) is a sequence of transitions \((q_0, s, q_1), (q_1, s, q_2), \ldots, (q_{n-1}, s, q_n)\) in \(\delta_t\), with \(n \geq 1\). For each \(t \in MOD \setminus EI(L)\), let \(A'_t\) be the automaton obtained from \(A_t\) by the following modification: for every \(s \in EI(L)\) and every \(s\)-path from state \(q_0\) to state \(q_n\) in \(A_t\), add the transition \((q_0, s, q_n)\) to \(A'_t\). We call the resulting automata \(A'_t\), for \(t \in MOD \setminus EI(L)\), the automata specifying \(eRG(L)\). It should be clear that, for \(t \in MOD \setminus EI(L)\), the word \(s_1 \ldots s_k\) is accepted by \(A'_t\) if \(s_1 \ldots s_k\) is derivable from \(t\) using the grammar \(eRG(L)\) since the modification simply adds “\(s\)-transitivity” for every \(s \in EI(L)\). Thus \(eRG(L)\) can be treated as a regular grammar for starting symbols outside \(EI(L)\).

Lemma 8. Let \(L\) be an \(eRG\) logic and \(t \in MOD \setminus EI(L)\). Then, for every \(n \geq 1\), and every \(s_1, s_2, \ldots, s_n \in MOD\), if the word \(s_1 \ldots s_n\) is derivable from \(t\) using the grammar \(eRG(L)\) then the formula \([t] \varphi \supset [s_1] \ldots [s_n] \varphi\) is \(L\)-valid.

Proof. This lemma can be proved by induction on the length of the derivation of \(s_1 \ldots s_k\) from \(t\) using the grammar \(eRG(L)\), analogously as for the case of regular grammar logics considered in Lemma 1. \(\square\)
For a set \( Q \) of states of automaton \( A \), the pair \((A, Q)\) can be treated as the automaton obtained from \( A \) by replacing the set of initial states by \( Q \). Thus, \( \mathcal{L}(A, Q) \) denotes the language generated by \((A, Q)\).

**LEMMA 9.** If \( L \) is an \( eRG \) logic, \( \{ A_t \mid t \in \mathcal{MOD} \setminus \mathcal{EI}(L) \} \) are the automata specifying \( eRG(L) \), \( s \in \mathcal{MOD} \setminus \mathcal{EI}(L) \), \( A_s = (\mathcal{MOD}, Q_s, I_s, \delta_s, F_s) \), and \( Q = \delta_s(I_s, \alpha_1) \cup \ldots \cup \delta_s(I_s, \alpha_h) \) for some words \( \alpha_1, \ldots, \alpha_h \) over alphabet \( \mathcal{MOD} \), then:

1. If \( t \rightarrow s_1 \ldots s_k \) is a rule of \( RG(L) \) then \( \mathcal{L}(A_s, \delta_s(Q, t)) \subseteq \mathcal{L}(A_s, \delta_s(Q, s_1 \ldots s_k)) \).

2. If \( t \in \mathcal{EI}(L) \) then \( \mathcal{L}(A_s, \delta_s(Q, t)) = \mathcal{L}(A_s, \tilde{\delta}_s(Q, tt)) \).

3. If \( \mathcal{L}(A_s, Q') \subseteq \mathcal{L}(A_s, Q'') \) then \( \mathcal{L}(A_s, \delta_s(Q', t)) \subseteq \mathcal{L}(A_s, \delta_s(Q'', t)) \).

**Proof.** Since \( \delta_s(Q' \cup Q'', t) = \delta_s(Q', t) \cup \delta_s(Q'', t) \) for all \( Q', Q'' \subseteq Q_s \), for assertions 1 and 2, we can assume \( Q = \delta_s(I_s, \alpha) \) for some word \( \alpha \).

1: Suppose \( \beta \) is a word over alphabet \( \mathcal{MOD} \), and \( \beta \in \mathcal{L}(A_s, \delta_s(Q, t)) \). Thus \( \alpha t \beta \in \mathcal{L}(A_s) \). If \( t \rightarrow s_1 \ldots s_k \) is a rule of \( RG(L) \), it follows that \( \alpha s_1 \ldots s_k \beta \in \mathcal{L}(A_s) \). Hence \( \beta \in \mathcal{L}(A_s, \delta_s(Q, s_1 \ldots s_k)) \).

2: Since \( t \rightarrow tt \) is a rule of \( RG(L) \) for all \( t \in \mathcal{EI}(L) \), the first assertion gives one half of the inclusion. It therefore suffices to show that \( \mathcal{L}(A_s, \delta_s(Q, tt)) \subseteq \mathcal{L}(A_s, \tilde{\delta}_s(Q, t)) \). Let \( \beta \in \mathcal{L}(A_s, \delta_s(Q, tt)) \). Thus \( \alpha t \beta \in \mathcal{L}(A_s) \). Because \( t \in \mathcal{EI}(L) \), we have \( tt \rightarrow t \) as a grammar rule of \( eRG(L) \). It follows that \( t \) is derivable from \( tt \) using the grammar \( eRG(L) \). Since \( A_s \) recognises the language derivable from \( s \) using \( eRG(L) \), it follows that \( \alpha t \beta \in \mathcal{L}(A_s) \). Hence \( \beta \in \mathcal{L}(A_s, \delta_s(Q, t)) \).

3: The third assertion clearly holds.

\[ \square \]

### 5. A Tableau Calculus for \( eRG \) Logics

Let \( L \) be an \( eRG \) logic and let \( \{ A_t \mid t \in \mathcal{MOD} \setminus \mathcal{EI}(L) \} \) be the automata specifying \( eRG(L) \). In the tableau calculus \( \mathcal{CL} \) given below, we use \( [t]Y \) to denote the set \( \{ [t] \varphi \mid \varphi \in Y \} \), and use \( \top \) to denote the truth constant with the usual semantics. Let \( \mathcal{CL} \) be the tableau calculus consisting of:

- rules \((\bot), (\land), (\lor)\) as in Table I,
rules \( (\text{label}) \), \( (\cup) \), \( (\text{add}) \), \( (\text{trans}) \) as in Table I with the additional condition \( t \notin EI(L) \) (we do not change the names of the rules),

- plus the following rules:

\[
(D) \quad \frac{X_t}{X; \langle t \rangle \top} \quad \text{if } t \in DI(L) \\
(5) \quad \frac{X; \langle t \rangle \varphi; [t](t) \varphi}{X; \langle t \rangle \varphi; [t][t] \varphi} \quad \text{if } t \in EI(L)
\]

\[
(\text{trans}_4) \quad \frac{X; [t] Y; \langle t \rangle \varphi}{\text{trans}(X, t); Y; [t] Y; \varphi} \quad \text{if } t \in EI(L)
\]

Rules \( (\text{trans}) \) and \( (\text{trans}_4) \) are transitional rules, and the other rules are static rules. Rule \( (\text{trans}_4) \) is a combination of rule \( (\text{trans}) \) and the traditional rule \( (4) \). Note that rules \( (\text{trans}) \) and \( (\text{trans}_4) \) of this calculus use the automata specifying \( eRG(L) \), not \( RG(L) \). This is a difference w.r.t. the calculus given for regular grammar logics.

Recall that \( \hat{S}_f(\varphi) \) denotes the set of all subformulae of \( \varphi \). let

\[
\tilde{S}_f^X = \bigcup_{\varphi \in X} S_f(\varphi) \cup \{ \bot, \top, \langle t \rangle \top | t \in \text{MOD} \}.
\]

Analogously as for regular grammar logics, our calculus has the analytic superformula property, with

\[
X_{CL} = \hat{S}_f(X) \cup \{ [A_t, Q] \varphi | t \notin EI(L), [t] \varphi \in \hat{S}_f(X) \text{ and } Q \subseteq Q_t \}
\cup \{ [t] \langle t \rangle \varphi | t \in EI(L) \text{ and } \langle t \rangle \varphi \in \hat{S}_f(X) \}.
\]

**Lemma 10.** The tableau calculus \( CL \) is sound.

**Proof.** The proof of this lemma is similar to the proof of Lemma 2. For proving soundness of rule \( (\text{label}) \), we need only to replace Lemma 1 by Lemma 8. Soundness of the other static rules is obvious. The proof of soundness of rule \( (\text{trans}) \) does not change. The proof for rule \( (\text{trans}_4) \) is similar to the proof for rule \( (\text{trans}) \), with an additional justification that \( [t] \psi \supset [t][t] \psi \) is an axiom of \( L \) when \( t \in EI(L) \). \( \square \)

### 5.1. Completeness

We assume that notions like “\( CL \)-saturation”, “saturated model graph”, and “consistent model graph” are defined analogously as for regular grammar logics.

For \( t \in EI(L) \) and \( \langle t \rangle \varphi \in X \), define

\[
\text{trans}_4(X, \langle t \rangle \varphi) = \text{trans}(X, t) \cup \{ \psi, [t] \psi | [t] \psi \in X \} \cup \{ \varphi \}.
\]

For \( t \in EI(L) \), define

\[
\text{core}_5(X, t) = \{ [t] \varphi | [t] \varphi \in X \} \cup \{ \langle t \rangle \varphi | \langle t \rangle \varphi \in X \} \cup
\}

\[
\{ [A_s, Q] \varphi | \exists \alpha, Q', [A_s, Q'] \varphi \in X \wedge Q = \delta_s(I_s, \alpha t) \subseteq Q' \}.
\]
As shown in the next lemma, $\text{core}_5(X, t)$ can be treated as the subset of $X$ consisting of formulae that are preserved when travelling through edges of $R_t$, including edges forced by the euclidean frame restriction.

**LEMMA 11.** Let $X$ be a $\mathcal{CL}$-saturation of some formula set and $Y$ be a $\mathcal{CL}$-saturation of $\text{trans}_4(X, \langle t \rangle \varphi)$ for some $\langle t \rangle \varphi \in X$ with $t \in EI(L)$. Then $\text{core}_5(X, t) \subseteq \text{core}_5(Y, t)$.

**Proof.** Due to the static rule (5), it suffices to show that if $[A_s, Q] \xi \in \text{core}_5(X, t)$ then $[A_s, Q] \xi \in \text{core}_5(Y, t)$. Suppose that $[A_s, Q] \xi \in \text{core}_5(X, t)$. Thus, there exist $\alpha$ and $Q'$ such that $Q = \delta_s(I_s, \alpha t) \subseteq Q'$ and $[A_s, Q'] \varphi \in X$. By definition of the set $\text{trans}_4$, there exists $[A_s, Q''] \varphi \in Y$ such that $\delta_s(Q', t) \subseteq Q''$. It follows that $\delta_s(I_s, \alpha tt) \subseteq Q''$. By Lemma 9, $\delta_s(I_s, \alpha tt) = \delta_s(I_s, att)$, hence $[A_s, Q] \xi \in \text{core}_5(Y, t)$. \quad $\square$

A $\mathcal{CL}$-consistent set $X$ is $\text{core}_5(t)$-saturated if for every $\langle t \rangle \varphi \in X$ and every $\mathcal{CL}$-saturation $Y$ of $\text{trans}_4(X, \langle t \rangle \varphi)$ we have $\text{core}_5(Y, t) = \text{core}_5(X, t)$.

Algorithm 3 given below for constructing an $L$-model graph of a finite $\mathcal{CL}$-consistent set $X$ differs from Algorithm 1 in the way it satisfies formulae of the form $\langle t \rangle \varphi$ with $t \in EI(L)$ at a world $w$. In Algorithm 3, for each $t \in EI(L)$: we find a $\text{core}_5(t)$-saturated set $U$ which is obtainable from $H(w)$ by applications of static $\mathcal{CL}$-rules and rule (trans$_4$) with the principal formula of the form $\langle t \rangle \psi$; we then create successors of $w$ via $R'_t$ to satisfy $\langle t \rangle$-formulae using $\text{core}_5(U, t)$ as the content of $w$. But we do this in two different ways depending upon whether $w$ has an $R'_t$-predecessor at this iteration. The intuitions for this dichotomy are based on the following insight from [9, Fig. 13] and [9, Pages 334-335]: the logic $\mathcal{KD}45$ is sound and complete w.r.t. the class of finite frames where each frame consists of a root which sees a possibly empty but finite strongly-connected-component or cluster.

To prove correctness of Algorithm 2, we use a data structure denoted by $\text{core}_5^\ast$ to store $\text{core}_5(U, t)$ in $\text{core}_5^\ast(w, t)$. Note that $\text{core}_5$ is a function, while $\text{core}_5^\ast$ is a table.

**ALGORITHM 2.**
Input: a finite $\mathcal{CL}$-consistent set $X$ of primitive language formulae. Output: an $L$-model graph $M = \langle W, \tau, \{R_t \mid t \in \text{MOD}\}, H \rangle$ of $X$.

1. Let $W = \{\tau\}$ and $R'_t = \emptyset$ for all $t \in \text{MOD}$.
   - Let $Y$ be a $\mathcal{CL}$-saturation of $X$ and let $H(\tau) = \text{compact}(Y)$.
   - Mark $\tau$ as unresolved.

2. While there are unresolved worlds, take one, say $w$, and do:
a) For every formula \( \langle t \rangle \varphi \) in \( H(w) \) with \( t \notin \mathcal{EI}(L) \):

i) Let \( U = \text{trans}(H(w), t) \cup \{ \varphi \} \) be the result of applying rule (trans) to \( H(w) \), let \( Y \) be a \( \mathcal{CL} \)-saturate of \( U \), and let \( Z = \text{compact}(Y) \).

ii) If \( \exists u \in W \) on the path from the root to \( w \) with \( H(u) = Z \), then add the pair \( (w, u) \) to \( R_t' \). Otherwise, add a new world \( u \) with content \( Z \) to \( W \), mark it as unresolved, and add the pair \( (w, u) \) to \( R_t' \).

b) For every \( t \in \mathcal{EI}(L) \) s.t. \( R_t'(v, w) \) does not hold for any \( v \):

i) Let \( U \) be a \( \mathcal{CL} \)-saturate of \( \text{trans}_4(H(w), \langle t \rangle \top) \).

ii) While there exist \( \langle t \rangle \varphi \in U \) and a \( \mathcal{CL} \)-saturate \( V \) of \( \text{trans}_4(U, \langle t \rangle \varphi) \) s.t. \( \text{core}_5(U, t) \subseteq \text{core}_5(V, t) \), let \( U = V \).

iii) Let \( \text{core}_5^5(w, t) = \text{core}_5(U, t) \).

iv) For every \( \langle t \rangle \varphi \in \text{core}_5^5(w, t) \):

- Let \( Y \) be a \( \mathcal{CL} \)-saturate of \( \text{trans}_4(\text{core}_5^5(w, t), \langle t \rangle \varphi) \).
- Let \( Z = \text{compact}(Y) \).
- Do the same as Step 2(a)ii.

c) For every \( t \in \mathcal{EI}(L) \) such that \( R_t'(v, w) \) holds for some \( v \):

Let \( \text{core}_5^5(w, t) = \text{core}_5(H(w), t) \).

d) Mark \( w \) as resolved.

3. Let \( R_t \) be the least extension of \( R_t' \) for \( t \in \mathcal{MOD} \) such that \( \langle W, \tau, \{ R_t \mid t \in \mathcal{MOD} \} \rangle \) is an \( L \)-frame.

This algorithm always terminates analogously to Algorithm 1.

**Lemma 12.** The following assertions are invariants during execution of Step 3 of the above algorithm (when the accessibility relations \( R_t \) for \( t \in \mathcal{MOD} \) are extended to \( R_t \)).

1. If \( R_t(w, u) \) holds and \( t \in \mathcal{EI}(L) \) then \( \text{core}_5^5(w, t) = \text{core}_5^5(u, t) = \text{core}_5(H(u), t) \).

2. If \( R_t(w, u) \) holds then for every formula \( [A_s, Q] \varphi \in H(w) \), there exists \( [A_s, Q'] \varphi \in H(u) \) such that \( L(A_s, \delta_s(Q, t)) \subseteq L(A_s, Q') \).

**Proof.** We first prove that if \( t \in \mathcal{EI}(L) \) then the first assertion implies the second one. As a consequence, we need to prove the second assertion only for the case \( t \notin \mathcal{EI}(L) \).

Suppose \( t \in \mathcal{EI}(L) \), that the first assertion holds, and \( [A_s, Q] \varphi \in H(w) \). Hence there exist words \( \alpha_1, \ldots, \alpha_k \) such that \( Q = \delta_s(I_s, \alpha_1) \cup \ldots \cup \delta_s(I_s, \alpha_k) \). By the computation of \( \text{core}_5^5(w, t) \),
we have \([A_s, \tilde{\delta}_s(I_s, \alpha_t)] \varphi \in \text{core}_x^s(w, t)\), for \(1 \leq i \leq k\). Hence \([A_s, \tilde{\delta}_s(I_s, \alpha_t)] \varphi \in \text{core}_x^s(H(u), t)\) for every \(1 \leq i \leq k\). It follows that there exists \([A_s, Q'] \varphi \in H(u)\) such that \(\delta_s(Q) \subseteq Q'\), and thus \(\mathcal{L}(A_s, \delta_s(Q, t)) \subseteq \mathcal{L}(A_s, Q')\).

We prove the assertions of the lemma by induction on the number of steps executed when extending \(R'_t\) for \(t \in \text{MOD}\) to \(R_t\).

Consider the base case, assume that \(t \in \mathcal{E}I(L)\). Hence \(u\) must have been created from \(w\) via Step 2b. We have that \(\text{core}^s_t(w, t) = \text{core}^s_t(H(u), t)\), because \(\text{core}^s_t(w, t)\) is \(\text{core}^s_t(t)\)-saturated and \(u\) is created from \(w\) via \(R'_t\) using \(\text{core}^s_t(w, t)\) as the content of \(w\). When \(u\) is resolved, we have that \(\text{core}^s_t(u, t) = \text{core}^s_t(H(u), t)\) due to Step 2c. Hence the first assertion holds. The second assertion clearly holds for the case \(t \notin \mathcal{E}I(L)\).

Consider the inductive step for the first assertion. If \(R_t(w, u)\) is created from \(R_t(w, v)\) and \(R_t(v, u)\) then, by the inductive assumption, \(\text{core}^s_t(w, t) = \text{core}^s_t(v, t)\) and \(\text{core}^s_t(v, t) = \text{core}^s_t(u, t) = \text{core}^s_t(H(u), t)\), which imply the first assertion. If \(R_t(w, u)\) is created from \(R_t(w, v)\) and \(R_t(v, u)\) then, by the inductive assumption, \(\text{core}^s_t(v, t) = \text{core}^s_t(w, t)\) and \(\text{core}^s_t(v, t) = \text{core}^s_t(u, t) = \text{core}^s_t(H(u), t)\), which imply the first assertion.

Consider the inductive step for the second assertion and the case when \(t \notin \mathcal{E}I(L)\). Suppose that \(R_t(w, u)\) is created from edges \(R_{s_i}(w_{i-1}, w_i)\) with \(1 \leq i \leq k\), \(w = w_0, u = w_k\), due to an inclusion \(R_{s_1} \circ \ldots \circ R_{s_k} \subseteq R_t\). Let \([A_s, Q] \varphi \in H(w)\). By Lemma 9(1), \(\mathcal{L}(A_s, \delta_s(Q, t)) \subseteq \mathcal{L}(A_s, \tilde{\delta}_s(Q, s_1 \ldots s_k))\). Let \(Q_0 = Q\). For \(i = 1, \ldots, k\), by the inductive assumption, there exists \([A_s, Q_i] \varphi \in H(w_i)\) such that \(\mathcal{L}(A_s, \delta_s(Q_{i-1}, s_i)) \subseteq \mathcal{L}(A_s, Q_i)\). For \(i = 2 \ldots k\), by Lemma 9(3), \(\mathcal{L}(A_s, \delta_s(Q, s_1 \ldots s_i)) \subseteq \mathcal{L}(A_s, Q_i)\) since \(\mathcal{L}(A_s, \delta_s(Q, s_1 \ldots s_{i-1})) \subseteq \mathcal{L}(A_s, Q_{i-1})\) and \(\delta_s(Q, s_1 \ldots s_i) = \delta_s(Q, s_1 \ldots s_{i-1}, s_i)\). Hence \(\mathcal{L}(A_s, \delta_s(Q, s_1 \ldots s_k)) \subseteq \mathcal{L}(A_s, Q_k)\). It follows that \(\mathcal{L}(A_s, \delta_s(Q, t)) \subseteq \mathcal{L}(A_s, Q_k)\). Choose \(Q' = Q_k\).

**LEMMA 13.** Let \(X\) be a finite \(CL\)-consistent set of formulae in the primitive language and \(M = \langle W, \tau, \{R_t \mid t \in \text{MOD}\}, H \rangle\) be the model graph for \(X\) constructed by Algorithm 2. Then \(M\) is a consistent saturated \(L\)-model graph satisfying \(X\).

**Proof.** It is clear that \(M\) is an \(L\)-model graph and for any \(w \in W\), the set \(H(w)\) is \(CL\)-consistent. We want to show that \(M\) is a saturated model graph. It suffices to show that:

1. For all \(w, u \in W\), if \([t] \varphi \in H(w)\) and \(R_t(w, u)\) holds then \(\varphi \in H(u)\).
2. For every \( w \in W \), if \( (t)\varphi \in H(w) \) and \( t \in \mathcal{EI}(L) \) then there exists \( u \in W \) such that \( R_t(w,u) \) holds and \( \varphi \in H(u) \).

For the first assertion, suppose \( [t]\varphi \in H(w) \) and \( R_t(w,u) \) holds.

Case \( t \notin \mathcal{EI}(L) \): Since \( [t]\varphi \in H(w) \), there exists \( [A_t,Q]\varphi \in H(w) \) with \( Q \supseteq I_t \). By Lemma 12, there exists \( [A_t,Q']\varphi \in H(u) \) such that \( \mathcal{L}(A_t,b_t(I_t,t)) \subseteq \mathcal{L}(A_t,Q') \). Since \( t \in \mathcal{L}(A_t) \), we have that \( \varepsilon \in \mathcal{L}(A_t,Q') \), which means \( Q' \cap F_t \neq \emptyset \).

Since \( [A_t,Q']\varphi \in H(u) \), it follows that \( \varphi \in H(u) \) by rule (add).

Case \( t \in \mathcal{EI}(L) \): Since \( [t]\varphi \in H(w) \), we have that \( [t]\varphi \in \text{core}_\varepsilon(w,t) \).

Since \( R_t(w,u) \) holds, there exists \( v \) such that \( R'_t(v,u) \) holds. By Lemma 12, \( \text{core}_\varepsilon(w,t) = \text{core}_\varepsilon(u,t) = \text{core}_\varepsilon(v,t) \). Hence \( [t]\varphi \in \text{core}_\varepsilon(v,t) \). Since \( R'_t(v,u) \) holds, it follows that \( \varphi \in H(u) \).

We now prove the second assertion. Suppose \( (t)\varphi \in H(w) \) and \( t \in \mathcal{EI}(L) \). If \( R'_t(v,w) \) does not hold for any \( v \) when \( w \) is resolved then \( w \) is connected via \( R'_t \) to a world \( u \) with \( \varphi \in H(u) \) at Step 2b since \( (t)\varphi \in \text{core}_\varepsilon(w,t) \). Alternatively, suppose \( R'_t(v,w) \) does hold for some \( v \) when \( w \) is resolved (at Step 2c). Since \( (t)\varphi \in H(w) \), we have \( (t)\varphi \in \text{core}_\varepsilon(H(w),t) = \text{core}_\varepsilon(v,t) \) by Lemma 12. Now \( v \) must have been considered at Step 2b in a previous iteration since this is the only way that an edge like \( R'_t(v,w) \) is created. Since \( (t)\varphi \in \text{core}_\varepsilon(v,t) \), this iteration must also create a world \( u \) with \( R'_t(v,u) \) such that \( \varphi \in H(u) \).

Then \( R_t(w,u) \) must hold after Step 3 by euclideaness.

The following theorem follows from Lemmas 10 and 13.

THEOREM 14. The calculus \( \mathcal{CL} \) for \( \mathcal{eRG} \) logics is sound and complete.

6. Complexity

Demri [4] proved that the general satisfiability problem of regular grammar logics is EXPTIME-complete by a transformation into PDL-satisfiability. We extend this result to \( \mathcal{eRG} \) logics using our tableau calculus.

Let \( L \) be an \( \mathcal{eRG} \) logic and \( X \) a finite formula set in the primitive language. Let \( n \) be the sum of the sizes of the formulae in \( X \) and the sizes of the automata specifying \( RG(L) \). Assume that \( n > |\mathcal{MOD}| \). To check whether \( X \) is \( L \)-satisfiable we can search for a closed \( \mathcal{CL} \)-tableau for \( X \), or equivalently, examine an and-or tree for \( X \) constructed by using Algorithm 2 to apply our \( \mathcal{CL} \)-tableau rules in a systematic way. In
such a tree, and-branching is caused by all possible applications of rules (trans) and (trans₄), while or-branching is caused by an application of rule (∨). The other CL-tableau rules are applied locally for each node whenever possible. There are at most \( n \) subformulae of \( X \) since \( X \) contains no automaton-modal operators, and there are at most \( 2^{O(n)} \) different automaton-modal operators. Due to the compact form, for each subformula \([t]\varphi\) of \( X \), a node contains at most one formula of the form \([A_t, Q]\varphi\). Thus, counting formulae generated by rule (5), a node contains at most \( (2^{O(n)})^{O(n)} = 2^{O(n^2)} \) different nodes.

Without rule (∪), there are at most \( 2^{2^{O(n)}} \) different nodes, breaking EXPTIME worst-case complexity, so the rule (∪) is absolutely essential. But it is not necessary for the satisfiability problem of a fixed logic.

Algorithm 2 terminates because it checks for repeated ancestors. Thus the same node can appear on multiple branches. In the worst case, Algorithm 2 therefore requires \( 2^{2^{O(n^2)}} \) time. We therefore refine it into Algorithm 3 below:

**ALGORITHM 3.**

Input: a finite set \( X \) of primitive language formulae and an eRG logic \( L \) with extended grammar specified by finite automata \( \{A_t \mid t \in \mathcal{MOD}\} \).

Output: a finite graph \( G = \langle V, E \rangle \).

1. Let \( G = \langle V, E \rangle = \langle \{ X \}, \emptyset \rangle \), and mark \( X \) as unresolved.

2. While \( V \) contains unresolved nodes, take one, say \( v \), and do:

   a) If (∨) is applicable to \( v \) then apply it to obtain denominators \( d_1 \) and \( d_2 \);

   b) Else if \( v \) is not closed w.r.t. a static rule of CL then apply the rule to \( v \) to obtain denominator \( d_1 \);

   c) Else, for every \( \langle t_i \rangle \varphi_i \) in \( v \) with \( t_i \notin E\mathcal{I}(L) \) [resp. \( t_i \in E\mathcal{I}(L) \)], let \( d_i = \text{trans}(v, t_i) \cup \{ \varphi_i \} \) [resp. \( d_i = \text{trans}_4(v, \langle t_i \rangle \varphi_i) \)] be the denominator obtained by applying (trans) [resp. (trans₄)] to \( v \).

   d) Mark \( v \) as resolved (\( v \) is an or/and node if the applied rule is/isn’t (∨)).

   e) For every denominator \( d_i \), \( 1 \leq i \leq k \):

      i) Let \( d = \text{compact}(d_i) \);

      ii) If some proxy \( c \in V \) has \( c = d \), add the edge \((v, c)\) to \( E \);

      iii) Else add \( d \) to \( V \), add \((v, d)\) to \( E \), and mark \( d \) as unresolved.
Algorithm 3 builds an and-or graph $G$ monotonically by “caching” previously seen nodes (but not their open/closed status). The graph $G$ contains a node $d$ for each denominator of an application of rule $(\lor)$, instead of a choice of a $CL$-saturation as in Algorithm 2. Each node appears only once because repetitions are represented by “cross-tree” edges to their first occurrence, so $G$ has at most $2^{O(n^2)}$ nodes.

We now make passes of the and-or graph $G$, marking nodes as $false$ in a monotonic way. In the first pass we mark the node containing $\bot$, if it exists, since $false$ captures inconsistency. In each subsequent pass we mark any unmarked or-node with two $false$-marked children, and mark any unmarked and-node with at least one $false$-marked child. We stop making passes when some pass marks no node. Otherwise, we must terminate after $2^{O(n^2)}$ passes since the root must then be marked with $false$. Note that once a node is marked with $false$ this mark is never erased. Finally, mark all non-$false$ nodes with $true$ giving graph $G_f$.

**Lemma 15.** If a node $v \in G_f$ is marked $false$ then $v$ is $CL$-inconsistent (and hence $L$-unsatisfiable).

*Proof.* By induction on the number of passes needed to mark $v$ with $false$. $\square$

**Lemma 16.** If a node $v \in G_f$ is marked $true$ then it is $L$-satisfiable.

*Proof.* Simulate Algorithm 2 for the formula set $v$, using $G_f$ as a guide. We guide Algorithm 2 to a $CL$-saturation of $v$ by choosing the denominator of each $(\lor')$ application which leads to $true$-marked nodes in $G_f$. When Algorithm 2 is required to create $R_t$-successors, we can again use $G_f$ to guide it to $true$-marked successors in $G_f$. In particular, if $t \in ET(L)$, then the $CL$-saturation of $v$ must contain $\langle t \rangle \top$ by rule (D). Thus Algorithm 3 must have a $true$-marked successor $u$ for $\langle t \rangle \top$. Although the arcs in $G_f$ are not labelled, we can create the arc $R'_t$ as required by Algorithm 2 since we know the value of $t$. A $CL$-saturation of $u$ can be found as above for $v$, giving the node $U$ required by Step 2(b)i of Algorithm 2.

All formula sets found by this simulation of Algorithm 2 are propositionally consistent because they are all marked $true$ in $G_f$, hence they do not contain $\bot$ or a pair $p$, $\neg p$. As for the proof of Lemma 13, it can be shown that the model graph constructed by the simulation is a consistent saturated $L$-model graph satisfying the formula set $v$. $\square$

Algorithm 3 and the creation of $G_f$ runs in time $(2^{O(n^2)})^2 = 2^{L.O(n^2)}$. Hence the general satisfiability problem of $eRG$ logics is in EXPTIME. In [4], Demri gave a regular grammar logic whose satisfiability problem is EXPTIME-hard. So, we arrive at:
THEOREM 17. The general satisfiability problem of eRG logics is EXPTIME-complete.

7. Further Work and Conclusions

Our main contributions are tableau calculi that form decision procedures for the class of regular grammar logics and the class of eRG logics. Our calculi also give a simple estimate of the upper complexity bound of regular grammar logics and eRG logics. In [10], it was shown that they can be used to obtain effective Craig interpolation for these logics. Our automaton-modal formulae are similar to formulae of APDL [11], but with a more compact representation using sets of states instead of single states. We have shown that automaton-modal formulae work well with the traditional techniques of proving soundness and completeness.

The class of regular grammar logics is large and mathematically interesting. However, these logics are not used in practice for reasoning about belief because they lack the seriality axioms for expressing consistency of belief. The class eRG of extended regular grammar logics contains useful epistemic logics from [17, 18] for reasoning about beliefs of agents like $KD_{4}^{(m)}$, $KD_{45}^{(m)}$, $KD_{14}$, $KD_{14s}$, $KD_{4Ig}$, and $KD_{4Ig5a}$. In particular, $KD_{4Ig5a}$ can be used to formalise the wise men puzzle [18]. This justifies our study of eRG logics.

Our restrictions on the class of extended regular grammar logics eRG are motivated mainly by technical difficulties in developing (non-prefixed) tableau calculi. The automata used in the CL calculus for an eRG logic $L$ specify the extended grammar $eRG(L)$ but not the regular grammar $RG(L)$. The technique used for proving completeness is also different from [10], as it is based on comparing automata/languages but not on a direct correspondence between inclusion axioms and grammar rules. Consequently, our extension to eRG logics is not at all trivial. For example, we currently see no way to add even the simple axiom $T : [t] \varphi \supset \varphi$ for E-indices. Note that eRG logics cannot be transformed into PDL because of axiom 5.

The prefixed tableaux of Baldoni et al. give a decision procedure only for right linear logics. A prefixed calculus that simulates our calculus would be less efficient because it would repeatedly search the current branch for computation, not just for loops as in our case. Moreover, it is well-known that loop checking can be done efficiently using, e.g., a hash table. Finally, the transformation of Demri and de Nivelle into GF$^2$ is based on states, but not sets of states, which reduces efficiency. Also their resulting formula sets are much larger because they keep a copy of
the formulae defining an automaton $A_t$ for each formula $[t] \varphi$, whereas we can keep only $t$ and $Q$ for $[A_t, Q]$ in $[A_t, Q] \varphi$. Similar observations have been stated for formulae of APDL.

Of course, efficient implementations require various optimisation techniques and heuristics, which are not mentioned here.

By propagating false “on the fly”, we believe we can prove global caching sound for checking satisfiability in multimodal $K$ with global assumptions i.e. “checking $ALC$-satisfiability of a concept w.r.t. a TBox with general axioms” [6].

References


