Analytic Cut-free Tableaux for Regular Modal Logics of Agent Beliefs

Rajeev Goré[1⋆] and Linh Anh Nguyen[2]

[1] The Australian National University and NICTA
Canberra ACT 0200, Australia
Rajeev.Gore@anu.edu.au
[2] Institute of Informatics, University of Warsaw
ul. Banacha 2, 02-097 Warsaw, Poland
nguyen@mimuw.edu.pl

Abstract. We present a sound and complete tableau calculus for a class $\mathcal{BREG}$ of extended regular grammar logics which contains useful epistemic logics for reasoning about agent beliefs. Our calculus is cut-free and has the analytic superformula property so it gives a decision procedure. Applying sound global caching to the calculus, we obtain the first optimal (EXPTime) tableau decision procedure for $\mathcal{BREG}$. We demonstrate the usefulness of $\mathcal{BREG}$ logics and our tableau calculus using the wise men puzzle and its modified version, which requires axiom (5) for single agents.

1 Introduction

Context-free grammar logics are normal multimodal logics characterised by "inclusion axioms" of the form $[t]\varphi \supset [s_1]\ldots[s_k]\varphi$, where $[t]$ and $[s_i]$ are modalities indexed by members t and s_i from some fixed set $\mathcal{MOD}$ of indices. Such logics are useful for modelling interactions between agents and groups of agents when indices from $\mathcal{MOD}$ denote agents/groups of agents. The general satisfiability problem of context-free grammar logics is undecidable [3], so researchers paid attention also to regular grammar logics, which are context-free grammar logics whose set of the corresponding grammar rules $t \to s_1 \ldots s_k$ forms either a left/right linear grammar or is specified by finite automata [3–5, 10]. Note that left/right linear grammars can be transformed in polynomial time to an equivalent finite automaton that is at most polynomially larger, and checking whether a context-free grammar generates a regular language is undecidable (see, e.g., [13]). To avoid ambiguity, we refer to regular grammar logics specified by finite automata as *regular modal logics*.

Regular grammar logics cannot be directly used for reasoning about belief due to the lack of axioms (D) and (5). It is commonly assumed that modal logics of belief invariably utilise the following:

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

⋆ NICTA is funded by the Australian Government's Dept of Communications, Information Technology and the Arts and the Australian Research Council through Backing Australia's Ability and the ICT Centre of Excellence program.

Positive Introspection: Axiom $(4) : [t]\varphi \supset [t][t]\varphi$ states that agents are aware of what they believe.

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

In [18], Nguyen studied a multimodal logic $KD4I_g5_a$ 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 $(I) : [t]\varphi \supset [s]\varphi$ for any (proper) super-group t of agent (group) s, but adopted axiom $(5) : \langle s\rangle\varphi \supset [s]\langle s\rangle\varphi$ 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 $[t]\varphi \supset [s]\varphi$ gives us $\langle s\rangle\varphi \supset \langle t\rangle\varphi$. If axiom (5) were present for the proper group t then $\langle s\rangle\varphi$ would give us $[t]\langle t\rangle\varphi$. But $\langle s\rangle\varphi \supset [t]\langle t\rangle\varphi$ states that the belief of a single agent s leads to a belief among the whole super-group t about φ. Conversely, the contra-positive $\langle t\rangle[t]\varphi \supset [s]\varphi$ states that if the group jointly does not believe that it does not jointly believe φ, then single agent s believes φ, which seems equally absurd. The logic $KD4I_g5_a$ formalises the most important properties of belief and common belief but does not give an exact formulation of common belief. It is similar to the well-known modal logic with common belief $KD45_n^C$ [12] and the modal logic with mutual belief [1] but it lacks the induction rule for common belief.

In this paper, we study the class $\mathcal{BR}eg$ of *regular modal logics of agent beliefs*, which are regular modal logics extended with axioms (D) and (5), where axiom (5) is adopted only for modal indices with the "terminal KD45-condition" as for the case of single agents in $KD4I_g5_a$ (see Section 2.2 for a formal definition). We extend our tableau calculus for regular modal logics [10] to a tableau calculus for $\mathcal{BR}eg$ logics. Our calculus for $\mathcal{BR}eg$ is sound, complete, cut-free and has the analytic superformula property. Applying sound global caching [10, 10] to it, we obtain the first optimal (EXPTime) tableau decision procedure for $\mathcal{BR}eg$.

The rest of this paper is structured as follows. In the end of this section, we present motivating examples and mention related works. In Section 2, we formally specify $\mathcal{BR}eg$ logics, introduce automaton-modal operators, and give definitions for tableau calculi. In Section 3, we present our tableau calculus for $\mathcal{BR}eg$, prove its soundness, and present "closed" tableaux for the motivating examples. In Section 4, we prove completeness of our tableau calculus. Due to the lack of space, an EXPTime decision procedure with global caching for $\mathcal{BR}eg$ is left in the Appendix. Section 5 concludes this work.

1.1 Motivating Examples

We will study two motivating examples about reasoning about beliefs and common beliefs of agents using our tableau calculus for $\mathcal{BR}eg$. The first one is the wise men puzzle, which is a famous benchmark introduced by McCarthy [17] for AI and has been previously studied in a considerable number of works (see [18] 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. 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 colour of his own card. Each wise man must announce the colour 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 $x \in \{a,b,c\}$, let $[x]\varphi$ stand for "the wise man x believes in φ" and let p_x stand for "the card of x is white". Let g denote the group $\{a,b,c\}$ and let $[g]$ informally stand for a certain operator of "common belief" of the group g. Let L_{wmp} be the $\mathcal{BRE}g$ logic which extends K_n ($n = 4$ and $\mathcal{MOD} = \{g,a,b,c\}$) with the following axioms:

$$[x]\varphi \supset \langle x\rangle\varphi \text{ and } [x]\varphi \supset [x][x]\varphi \quad \text{for } x \in \{g,a,b,c\},$$
$$[g]\varphi \supset [x]\varphi \text{ and } \langle x\rangle\varphi \supset [x]\langle x\rangle\varphi \quad \text{for } x \in \{a,b,c\}.$$

The wise men puzzle can be formalised as follows:

– If y sits behind x then either x's card is white or y knows that x's card is not white:

$$\varphi_1 = [g](p_a \vee [b]\neg p_a) \qquad \varphi_2 = [g](p_a \vee [c]\neg p_a) \qquad \varphi_3 = [g](p_b \vee [c]\neg p_b)$$

– At least one of the men has a white card:

$$\varphi_4 = [g](p_a \vee p_b \vee p_c)$$

– Each of b and c does not know the colour of his own card. In particular, each of the men considers that it is possible that his own card is not white:

$$\varphi_5 = [g]\langle b\rangle\neg p_b \qquad \varphi_6 = [g]\langle c\rangle\neg p_c$$

– The question is whether a believes that his card is white ($[a]p_a$). That is, whether $(\varphi_1 \wedge \ldots \wedge \varphi_6) \supset [a]p_a$ is L_{wmp}-valid. This is equivalent to whether the formula set $\Gamma_{wmp} = \{\varphi_1, \ldots, \varphi_6, \langle a\rangle\neg p_a\}$ is L_{wmp}-unsatisfiable.

As we will see, the wise men puzzle is solvable in a regular modal logic without axioms (D) and (5). More specifically, Γ_{wmp} is unsatisfiable in the logic L'_{wmp} obtained from L_{wmp} by discarding axioms (D) and (5). So, we introduce a modified version of the wise men puzzle for which axiom (5) is necessary[3]: "The wise men sit in the order a, b, c, all facing the same direction. 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 as soon as b or c knows that his own card is white or that the card of the man behind is white he must inform the man in the front about that, and as soon as a knows who has a white card he must announce that. The question is whether a will know who has a white card." To formulate the new puzzle, we discard the formulae φ_5 and φ_6 and add to the formula set the following formulae:

$$\varphi'_5 = [g]([c]p_c \supset [b]p_c) \qquad \varphi'_6 = [g]([b]p_b \supset [a]p_b) \qquad \varphi'_7 = [g]([b]p_c \supset [a]p_c)$$

The new formula set is thus $\Delta_1 = \{\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi'_5, \varphi'_6, \varphi'_7\}$. The question is whether $\Delta_1 \supset [a]p_a \vee [a]p_b \vee [a]p_c$ is L_{wmp}-valid, or equivalently, whether $\Delta_{wmp} = \Delta_1 \cup \{\langle a\rangle\neg p_a, \langle a\rangle\neg p_b, \langle a\rangle\neg p_c\}$ is L_{wmp}-unsatisfiable.

[3] It can be shown that the formula set Δ_{wmp} given below is L'_{wmp}-satisfiable.

1.2 Related Works

In the previous work [10], we gave an analytic tableau calculus for regular modal logics and presented an EXPTime decision procedure for such logics. The class $\mathcal{BR}eg$ studied in this paper extends the class of regular modal logics with useful epistemic logics for reasoning about agent beliefs.

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. Adding axiom (D) to regular grammar logics with converse results in a class larger than $\mathcal{BR}eg$. We cannot compare efficiency of the two approaches (for $\mathcal{BR}eg$) yet, but our tableau decision procedure for $\mathcal{BR}eg$ is certainly worth studying and experimenting. (Demri and de Nivelle wrote *"It is too early to state that the transformation from regular grammar logics with converse into GF^2 defined in this paper can be used to mechanize efficiently such source logics with a prover for GF^2..."* [5, page 293].)

Other related works are works on regular grammar logics [3, 4], works on PDL-like logics (e.g. [11, 5, 2]), and works on epistemic logics (e.g. [12, 1]). However, the first two groups often lack axioms (D) and (5) and are not devoted to reasoning about epistemic states of agents, while the third group often adopts only some specific axioms but not the wide range of inclusion axioms. The class of "incestual multimodal logics" studied by Baldoni in [3] is large and contains the class $\mathcal{BR}eg$, but the general satisfiability problem for it is undecidable.

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 and q for elements 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:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \supset \varphi \mid [t]\varphi \mid \langle t\rangle\varphi.$$

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 current world, and each R_t is a binary relation on W, called the accessibility relation for $[t]$ and $\langle t\rangle$. 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 $\langle W, \tau, (R_t)_{t\in\mathcal{MOD}}, h\rangle$, 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. We sometimes treat model graphs as models with the range of H restricted to $\mathcal{PROP}$.

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:

$$M,w \models [t]\varphi \quad \text{iff} \quad \forall v \in W.\ R_t(w,v) \text{ implies } M,v \models \varphi$$
$$M,w \models \langle t\rangle\varphi \quad \text{iff} \quad \exists v \in W.\ R_t(w,v) \text{ and } M,v \models \varphi.$$

We say that φ *is satisfied at w in M* if $M,w \models \varphi$. We say that *M satisfies* φ and call *M* a *model of* φ if $M,\tau \models \varphi$.

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, if not stated otherwise, 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 φ has a logically equivalent formula φ' which is in negation normal form. We treat a finite set of formulae as the conjunction of its formulae.

2.2 A Class $\mathcal{BR}eg$ of Regular Modal Logics of Agent Beliefs

A $\mathcal{BR}eg$ logic is a normal multimodal logic L extending K_n with:

Inclusion Axioms: a set $IA(L)$ of inclusion axioms $[t]\varphi \supset [s_1]\dots[s_k]\varphi$ with $k \geq 0$ whose corresponding grammar rules $t \to s_1 \dots s_k$ jointly form a grammar $RG(L)$ specified by finite automata $(A_s)_{s\in\mathcal{M}\ o\mathcal{D}}$ such that A_s for $s \in \mathcal{M}\ o\mathcal{D}$ recognises the set of words derivable from s using the rules of $RG(L)$;[4]

Seriality Axioms: a set $\mathcal{D}I(L) \subseteq \mathcal{M}\ o\mathcal{D}$ of D-indices with corresponding seriality axioms $[t]\varphi \supset \langle t\rangle\varphi$ for every $t \in \mathcal{D}I(L)$;

Terminal KD45-Condition: a set $\mathcal{E}I(L) \subseteq \mathcal{D}I(L)$ of E-indices with corresponding axioms $[t]\varphi \supset [t][t]\varphi \in IA(L)$ and $\langle t\rangle\varphi \supset [t]\langle t\rangle\varphi$ for every $t \in \mathcal{E}I(L)$ and the condition that $IA(L)$ contains no other inclusion axioms of the form $[t]\varphi \supset [s_1]\dots[s_k]\varphi$ for $t \in \mathcal{E}I(L)$.

Recall that a *finite automaton A* is a tuple $\langle \Sigma, Q, I, \delta, F\rangle$, where: Σ is the alphabet (for our case $\Sigma = \mathcal{M}\ o\mathcal{D}$); 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/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.

The *L-frame restrictions* for a $\mathcal{BR}eg$ logic L are the following restrictions:

- $R_{s_1} \circ \dots \circ R_{s_k} \subseteq R_t$ if $s_1 \dots s_k$ is accepted by A_t, where $t \in \mathcal{M}\ o\mathcal{D}$ and $(A_s)_{s\in\mathcal{M}\ o\mathcal{D}}$ are the finite automata specifying $RG(L)$;
- R_t is serial (i.e. $\forall u \exists w R_t(u,w)$) for each D-index $t \in \mathcal{D}I(L)$;
- R_t is transitive and euclidean (i.e. $\forall u,v,w R_t(u,v) \wedge R_t(v,w) \to R_t(u,w)$ and $\forall u,v,w R_t(u,v) \wedge R_t(u,w) \to R_t(w,v)$) for each E-index $t \in \mathcal{E}I(L)$.

[4] If $k = 0$ then the right hand side of $t \to s_1 \dots s_k$ stands for the empty word ε.

We do not require axiom (D) for every modal index in $\mathcal{BR}eg$ logics, but we allow axiom (5) only for modal indices which satisfy the terminal KD45-condition. This restriction can be justified from practical considerations stated for $KD4I_g5_a$ in Introduction. It can be shown that the multimodal logics of belief $KDI4$, $KDI4_s$, $KD4I_g$, $KD4I_g5_a$ studied by Nguyen in [20, 18], as well as $KD45_{(m)}$, belong to $\mathcal{BR}eg$.

A Kripke model is an *L-model* if its frame satisfies all L-frame restrictions. A formula φ is *L-satisfiable* if there exists an L-model satisfying it. A formula φ is *L-valid* if every L-model satisfies it.

It can be shown that for a $\mathcal{BR}eg$ logic L, a formula φ is L-valid iff it is derivable using the axiomatisation of L. (See [20] for the correspondence theory.)

2.3 Some Properties of $\mathcal{BR}eg$ Logics

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

Let L be a $\mathcal{BR}eg$ logic and let $(A_t)_{t\in\mathcal{M}\,\mathcal{OD}}$ 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-2},s,q_{n-1}), (q_{n-1},s,q_n) in δ_t, with $n \geq 1$. For each $t \in \mathcal{M}\,\mathcal{OD} \setminus \mathcal{EI}(L)$, let A'_t be the automaton obtained from A_t by the following modification: for every $s \in \mathcal{EI}(L)$ 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 $t \in \mathcal{M}\,\mathcal{OD} \setminus \mathcal{EI}(L)$, the *automata specifying* $eRG(L)$. It should be clear that, for $t \in \mathcal{M}\,\mathcal{OD} \setminus \mathcal{EI}(L)$, the word $s_1 \ldots s_k$ is accepted by A'_t iff $s_1 \ldots s_k$ is derivable from t using the grammar $eRG(L)$ since the modification simply adds "s-transitivity" for every $s \in \mathcal{EI}(L)$. Thus $eRG(L)$ can be treated as a regular grammar for starting symbols outside $\mathcal{EI}(L)$.

The logic L_{wmp} specified in Introduction is a $\mathcal{BR}eg$ logic with $\mathcal{DI}(L_{wmp}) = \{g,a,b,c\}$, $\mathcal{EI}(L_{wmp}) = \{a,b,c\}$, and the extended grammar $eRG(L_{wmp})$ specified by the following finite automaton:

$$A_g = \langle \mathcal{M}\,\mathcal{OD}, \{0,1\}, \{0\}, \{(0,x,0),(0,x,1) \mid x \in \mathcal{M}\,\mathcal{OD}\}, \{1\}\rangle$$

Lemma 1. *Let L be a $\mathcal{BR}eg$ logic and $t \in \mathcal{M}\,\mathcal{OD} \setminus \mathcal{EI}(L)$. Then, for every $n \geq 1$, and every $s_1, s_2, \cdots, s_n \in \mathcal{M}\,\mathcal{OD}$, 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. By induction on the length of the derivation of $s_1 \ldots s_k$ from t using $eRG(L)$.

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).

Let $A = \langle \mathcal{M}\,\mathcal{OD}, Q_A, I_A, \delta_A, F_A\rangle$ and let ε denote the empty word. For $Q \subseteq Q_A$, $t \in \mathcal{M}\,\mathcal{OD}$, and a word α over alphabet $\mathcal{M}\,\mathcal{OD}$, define: $\delta_A(Q,t) = \{q' \mid \exists q \in Q.(q,t,q') \in \delta_A\}$, $\widetilde{\delta}_A(Q,\varepsilon) = Q$, $\widetilde{\delta}_A(Q,\alpha t) = \delta_A(\widetilde{\delta}_A(Q,\alpha),t)$. For $A_s = \langle \mathcal{M}\,\mathcal{OD}, Q_s, I_s, \delta_s, F_s\rangle$, we write δ_s (resp. $\widetilde{\delta}_s$) instead of δ_{A_s} (resp. $\widetilde{\delta}_{A_s}$).

Lemma 2. *Let L be a $\mathcal{BR}eg$ logic, $(A_t)_{t\in\mathcal{MOD}\setminus\mathcal{EI}(L)}$ the automata specifying $eRG(L)$, $s\in\mathcal{MOD}\setminus\mathcal{EI}(L)$, $A_s=\langle\mathcal{MOD},Q_s,I_s,\delta_s,F_s\rangle$, and $Q=\widetilde{\delta_s}(I_s,\alpha_1)\cup\ldots\cup\widetilde{\delta_s}(I_s,\alpha_h)$ for some words $\alpha_1,\ldots\alpha_h$ over alphabet $\mathcal{MOD}$. Then:*

1. *If $t\to s_1\ldots s_k$ is a rule of $RG(L)$ then $\mathcal{L}(A_s,\widetilde{\delta_s}(Q,t))\subseteq\mathcal{L}(A_s,\widetilde{\delta_s}(Q,s_1\ldots s_k))$.*
2. *If $t\in\mathcal{EI}(L)$ then $\mathcal{L}(A_s,\widetilde{\delta_s}(Q,t))=\mathcal{L}(A_s,\widetilde{\delta_s}(Q,tt))$.*
3. *If $\mathcal{L}(A_s,Q')\subseteq\mathcal{L}(A_s,Q'')$ then $\mathcal{L}(A_s,\widetilde{\delta_s}(Q',t))\subseteq\mathcal{L}(A_s,\widetilde{\delta_s}(Q'',t))$.*

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 $Q=\widetilde{\delta_s}(I_s,\alpha)$ for some word α.

1: Suppose β is a word over alphabet $\mathcal{MOD}$, and $\beta\in\mathcal{L}(A_s,\widetilde{\delta_s}(Q,t))$. Thus $\alpha t\beta\in\mathcal{L}(A_s)$. If $t\to s_1\ldots s_k$ is a rule of $RG(L)$, it follows that $\alpha s_1\ldots s_k\beta\in\mathcal{L}(A_s)$. Hence $\beta\in\mathcal{L}(A_s,\widetilde{\delta_s}(Q,s_1\ldots s_k))$.
2: Since $t\to tt$ 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 $\mathcal{L}(A_s,\widetilde{\delta_s}(Q,tt))\subseteq\mathcal{L}(A_s,\widetilde{\delta_s}(Q,t))$. Let $\beta\in\mathcal{L}(A_s,\widetilde{\delta_s}(Q,tt))$. Thus $\alpha tt\beta\in\mathcal{L}(A_s)$. Because $t\in\mathcal{EI}(L)$, we have $tt\to t$ 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 $\beta\in\mathcal{L}(A_s,\widetilde{\delta_s}(Q,t))$.
3: The third assertion clearly holds.

2.4 Automaton-Modal Formulae

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. Similar constructions were previously used in [11, 14, 10, 21].

Given a Kripke model $M=\langle W,\tau,(R_t)_{t\in\mathcal{MOD}},h\rangle$ and $w_0\in W$, define that $M,w_0\models[A,Q]\varphi$ if $M,w_k\models\varphi$ for every path $w_0R_{s_1}w_1\ldots w_{k-1}R_{s_k}w_k$ with $k\geq 0$ and $\widetilde{\delta_A}(Q,s_1\ldots s_k)\cap F_A\neq\emptyset$ (i.e. $s_1\ldots s_k$ is accepted by A when starting from some state from Q).

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. Note that an automaton-modal operator can appear only at the beginning of a formula.

2.5 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* σ 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* $\mathcal{C}L$ for a logic L is a finite set of tableau rules.

$$(\bot)\ \frac{X;p;\neg p}{\bot} \qquad (\wedge)\ \frac{X;\varphi\wedge\psi}{X;\varphi\wedge\psi;\varphi;\psi} \qquad (\vee)\ \frac{X;\varphi\vee\psi}{X;\varphi\vee\psi;\varphi \mid X;\varphi\vee\psi;\psi}$$

$$(D)\ \frac{X}{X;\langle t\rangle\top}\ \text{if } t\in \mathcal{D}\,I(L) \qquad (5)\ \frac{X;\langle t\rangle\varphi}{X;\langle t\rangle\varphi;[t]\langle t\rangle\varphi}\ \text{if } t\in \mathcal{E}\,I(L)$$

$$(\mathsf{aut})\ \frac{X;[t]\varphi}{X;[t]\varphi;[A_t,I_t]\varphi}\ \text{if } t\notin \mathcal{E}\,I(L) \qquad (\mathsf{add})\ \frac{X;[A_t,Q]\varphi}{X;[A_t,Q]\varphi;\varphi}\ \text{if } Q\cap F_t\neq\emptyset$$

$$(\mathsf{trans})\ \frac{X;\langle t\rangle\varphi}{\mathsf{trans}(X,t);\varphi}\ \text{if } t\notin \mathcal{E}\,I(L) \qquad (\mathsf{trans}_4)\ \frac{X;[t]Y;\langle t\rangle\varphi}{\mathsf{trans}(X,t);Y;[t]Y;\varphi}\ \text{if } t\in \mathcal{E}\,I(L)$$

Table 1. Tableau Rules for $\mathcal{BR}eg$ Logics

A $\mathcal{C}L$-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 Δ be a set of tableau rules. We say that Y is *obtainable from X by applications of rules from* Δ if there exists a tableau for X which uses only rules from Δ and has a node that carries Y. A node to which 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 is $\mathcal{C}L$-*consistent* if every $\mathcal{C}L$-tableau for X is open. If there is a closed $\mathcal{C}L$-tableau for X then X is $\mathcal{C}L$-*inconsistent*.

A tableau calculus $\mathcal{C}L$ is *sound* if for all finite formula sets X in the primitive language, X is L-satisfiable implies X is $\mathcal{C}L$-consistent. It is *complete* if for all finite formula sets X in the primitive language, X is $\mathcal{C}L$-consistent implies X is L-satisfiable. We say that a rule σ of $\mathcal{C}L$ is sound w.r.t. L if for every instance σ' of σ, if the numerator of σ' is L-satisfiable then so is one of the denominators of σ'. Any calculus $\mathcal{C}L$ containing only rules sound w.r.t. L is sound.

3 A Tableau Calculus for $\mathcal{BR}eg$ Logics

Fix a $\mathcal{BR}eg$ logic L and let $(A_t = \langle \mathcal{M}^{\mathcal{O}\mathcal{D}}, Q_t, I_t, \delta_t, F_t\rangle)_{t\in \mathcal{M}^{\mathcal{O}\mathcal{D}}\setminus \mathcal{E}\,I(L)}$ be the automata specifying $eRG(L)$. Recall that formulae are in negation normal form. We use X, Y to denote formula sets, use $[t]X$ to denote the set $\{[t]\varphi \mid \varphi\in X\}$, and use $\top$ to denote the truth constant with the usual semantics. We write $X;Y$ for $X\cup Y$, write $X;\varphi$ for $X\cup\{\varphi\}$, and $\varphi;\psi$ for $\{\varphi,\psi\}$.

The transfer of X through $\langle t\rangle$, denoted by $\mathsf{trans}(X,t)$, is:

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

The tableau calculus $\mathcal{C}L$ is given in Table 1. The last two rules (trans) and (trans_4) are *transitional* rules, while the remaining rules except $(\bot)$ are *static* rules. The intuition

of this sorting is that static rules keep us in the same world of the Kripke model under construction, while transitional rules take us to a new Kripke successor world.

Note that we include the principal formula of the static rules in their denominators.[5] Thus, the numerator of any static rule is a subset of every one of its denominators. 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. We implicitly assume that a static rule is applied to X only when X is not closed w.r.t. that rule and treat this as an (additional) condition for applying the rule.

A tableau calculus $\mathcal{C}$ has the *analytic superformula* property iff to every finite set X we can assign a finite set $X^*_{\mathcal{C}}$ which contains all formulae that may appear in any tableau for X. We write $Sf(\varphi)$ for the set of all subformulae of φ, and $Sf(X)$ for the set $\bigcup_{\varphi\in X} Sf(\varphi)\cup\{\bot\}$. Our calculus $\mathcal{C}L$ has the analytic superformula property, with

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

Lemma 3. *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 $M = \langle W,\tau,(R_t)_{t\in\mathcal{M}\,\mathcal{O}\mathcal{D}},h\rangle$. 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 rules (trans) and (trans_4), we show that its denominator is satisfied at some world reachable from w via R_t in the same L-model.

$(\bot),(\wedge),(\vee),(D),(5)$**:** These cases are obvious.

(aut)**:** 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 $1\le i\le k$ 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, $M,w\models[A_t,I_t]\varphi$.

(add)**:** This case follows from the semantics of automaton-modal formulae.

(trans)**:** Suppose that $M,w\models X;\langle t\rangle\varphi$. 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.

(trans_4)**:** The proof for this case is similar to the proof for the case of (trans), with an additional justification that $[t]\psi\supset[t][t]\psi$ is an axiom of L when $t\in\mathcal{EI}(L)$.

3.1 Examples

In this subsection, we present closed tableaux for the formula sets formalising the wise men puzzle. Let L be the $\mathcal{BR}eg$ logic L_{wmp} defined in Introduction. Recall that the following automaton A_g specifies $eRG(L)$:

$$A_g = \langle\mathcal{M}\,\mathcal{O}\mathcal{D},\{0,1\},\{0\},\{(0,x,0),(0,x,1)\mid x\in\mathcal{M}\,\mathcal{O}\mathcal{D}\},\{1\}\rangle$$

In Figure 3.1, we give a closed $\mathcal{C}L$-tableau for the formula set Γ_{wmp}, which was specified in Introduction for formalising the wise men puzzle. In that tableau, for $1\le i\le 6$, φ_i is the formula as in Introduction, ψ_i is the subformula of φ_i such

[5] This allows an easier proof for soundness of global caching.

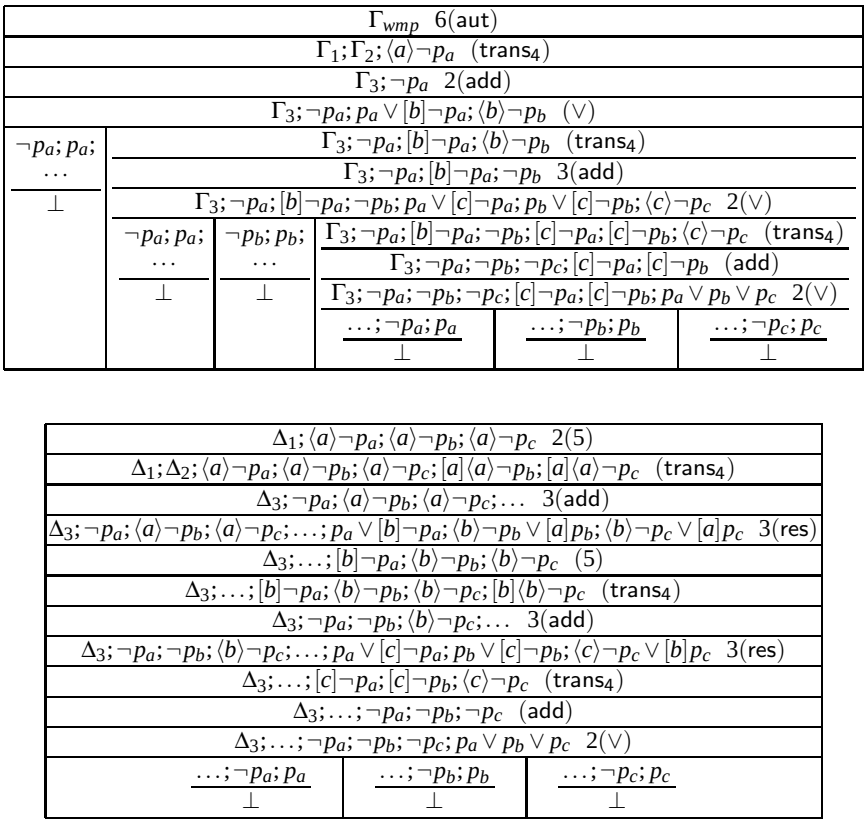

Fig. 1. Closed $\mathcal{C}L$-Tableaux for the Wise Men Puzzle

that $\varphi_i = [g]\psi_i$, $\Gamma_1 = \{\varphi_1,\ldots,\varphi_6\}$, $\Gamma_2 = \{[A_g,\{0\}]\psi_1,\ldots,[A_g,\{0\}]\psi_6\}$, and $\Gamma_3 = \{[A_g,\{0,1\}]\psi_1,\ldots,[A_g,\{0,1\}]\psi_6\}$. Since the tableau calculus $\mathcal{C}L$ is sound, it follows that Γ_{wmp} is L-unsatisfiable.

For the modified version of the wise men puzzle, let $\varphi_1,\ldots,\varphi_4,\varphi'_5,\varphi'_6,\varphi'_7$ be the formulae as in Introduction. In the negation normal form, we have that:

$$\varphi'_5 = [g](\langle c\rangle\neg p_c\vee[b]p_c) \qquad \varphi'_6 = [g](\langle b\rangle\neg p_b\vee[a]p_b) \qquad \varphi'_7 = [g](\langle b\rangle\neg p_c\vee[a]p_c)$$

For $1\leq i\leq 7$, let ψ_i be the formula such that $\varphi_i = [g]\psi_i$ if $1\leq i\leq 4$, and $\varphi'_i = [g]\psi_i$ if $i\in\{5,6,7\}$. Let $\Delta_1 = \{\varphi_1,\varphi_2,\varphi_3,\varphi_4,\varphi'_5,\varphi'_6,\varphi'_7\}$, $\Delta_2 = \{[A_g,\{0\}]\psi_i \mid 1\leq i\leq 7\}$, and $\Delta_3 = \{[A_g,\{0,1\}]\psi_i \mid 1\leq i\leq 7\}$. Let $\overline{\varphi}$ denote the negation normal form of $\neg\varphi$. Note that the following rule is "derivable" using the rules of $\mathcal{C}L$:

$$(\mathsf{res})\ \frac{X;\varphi\vee\psi;\overline{\varphi}}{X;\varphi\vee\psi;\overline{\varphi};\psi} \quad\text{or}\quad \frac{X;\psi\vee\varphi;\overline{\varphi}}{X;\psi\vee\varphi;\overline{\varphi};\psi}$$

In Figure 3.1, we also give a closed $\mathcal{C}L$-tableau using the tableau rule (res) for the formula set $\Delta_{wmp} = \Delta_1\cup\{\langle a\rangle\neg p_a,\langle a\rangle\neg p_b,\langle a\rangle\neg p_c\}$, which was specified in Introduction

for formalising the modified version of the wise men puzzle. Since the tableau calculus $\mathcal{C}L$ is sound, it follows that Δ_{wmp} is L-unsatisfiable.

4 Completeness

4.1 Proving Completeness via Model Graphs

Let L be a $\mathcal{BR}eg$ logic. We prove completeness of our calculus via model graphs following [19, 9, 15, 16, 10] by giving an algorithm that accepts a finite $\mathcal{C}L$-consistent formula set X in the primitive language and constructs an L-model graph (defined below) for X that satisfies each of its formulae at the appropriate world.

For a finite $\mathcal{C}L$-consistent formula set X, a formula set Y is called a *$\mathcal{C}L$-saturation of X* if Y is a maximal $\mathcal{C}L$-consistent set obtainable from X by applications of the static rules of $\mathcal{C}L$.

Lemma 4. *Let X be a finite $\mathcal{C}L$-consistent formula set and Y a $\mathcal{C}L$-saturation of X. Then $X \subseteq Y \subseteq X^*_{\mathcal{C}L}$ and Y is closed w.r.t. the static rules of $\mathcal{C}L$. Furthermore, there is an effective procedure that, given a finite $\mathcal{C}L$-consistent formula set X, constructs some $\mathcal{C}L$-saturation of X.*

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

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

A model graph is an *L-model graph* if its frame is an L-frame. An L-model graph $\langle W, \tau, (R_t)_{t \in \mathcal{M}_{\mathcal{OD}}}, H \rangle$ is *saturated* if every $w \in W$ satisfies:

- if $\varphi \wedge \psi \in H(w)$ then $\{\varphi, \psi\} \subseteq H(w)$;
- if $\varphi \vee \psi \in H(w)$ 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 $\langle t \rangle \varphi \in H(w)$ 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 5. *If $M = \langle W, \tau, (R_t)_{t \in \mathcal{M}_{\mathcal{OD}}}, H \rangle$ 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 φ.

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

4.2 Constructing Model Graphs

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

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

Observe that the compact form does not affect the essence of $\mathcal{C}L$-tableaux. More specifically, if applying a $\mathcal{C}L$-tableau rule to X gives denominators $Y_1, \ldots, Y_k$, then applying that rule to $\mathsf{compact}(X)$ gives denominators $Z_1, \ldots, Z_k$ such that $\mathsf{compact}(Z_i) = \mathsf{compact}(Y_i)$ for $1 \leq i \leq k$. In particular, "compacting" preserves $\mathcal{C}L$-consistency and $\mathcal{C}L$-inconsistency.

For $t \in \mathcal{E}\mathit{I}(L)$ and $\langle t \rangle \varphi \in X$, define

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

For $t \in \mathcal{E}\mathit{I}(L)$, define

$$\begin{aligned}\mathsf{core}_5(X, t) = {} & \{[t]\varphi \mid [t]\varphi \in X\} \cup \{\langle t \rangle \varphi \mid \langle t \rangle \varphi \in X\} \cup \\ & \{[A_s, Q]\varphi \mid \exists \alpha, Q'.[A_s, Q']\varphi \in X \text{ and } Q = \widetilde{\delta_s}(I_s, \alpha t) \subseteq Q'\}.\end{aligned}$$

As shown in the next lemma, $\mathsf{core}_5(X, t)$ 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 6. *Let X be a $\mathcal{C}L$-saturation of some formula set and Y be a $\mathcal{C}L$-saturation of $\mathsf{trans}_4(X, \langle t \rangle \varphi)$ for some $\langle t \rangle \varphi \in X$ with $t \in \mathcal{E}\mathit{I}(L)$. Then $\mathsf{core}_5(X, t) \subseteq \mathsf{core}_5(Y, t)$.*

Proof. Due to the static rule (5), it suffices to show that if $[A_s, Q]\xi \in \mathsf{core}_5(X, t)$ then $[A_s, Q]\xi \in \mathsf{core}_5(Y, t)$. Suppose that $[A_s, Q]\xi \in \mathsf{core}_5(X, t)$. Thus, there exist α and Q' such that $Q = \widetilde{\delta_s}(I_s, \alpha t) \subseteq Q'$ and $[A_s, Q']\varphi \in X$. By definition of the set trans_4, there exists $[A_s, Q'']\varphi \in Y$ such that $\delta_s(Q', t) \subseteq Q''$. It follows that $\widetilde{\delta_s}(I_s, \alpha t t) \subseteq Q''$. By Lemma 2, $\widetilde{\delta_s}(I_s, \alpha t) = \widetilde{\delta_s}(I_s, \alpha t t)$, hence $[A_s, Q]\xi \in \mathsf{core}_5(Y, t)$.

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

Algorithm 1 given below constructs a consistent saturated L-model graph for a finite $\mathcal{C}L$-consistent set X. In this algorithm, for each $t \in \mathcal{E}\mathit{I}(L)$: we find a $\mathsf{core}_5(t)$-saturated set U which is obtainable from $H(w)$ by applications of static $\mathcal{C}L$-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 $\mathsf{core}_5(U, t)$ 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 *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 1, we use a data structure denoted by core_5^* to store $\mathsf{core}_5(U, t)$ in $\mathsf{core}_5^*(w, t)$. Note that core_5 is a function, while core_5^* is a table. In the algorithm, the worlds of the constructed model graph are marked either as *unresolved* or as *resolved*.

Algorithm 1
Input: a finite $\mathcal{C}L$-consistent set X of primitive language formulae.
Output: an L-model graph $M = \langle W, \tau, (R_t)_{t \in \mathcal{M}^{\mathcal{OD}}}, H \rangle$ of X.

1. Let $W = \{\tau\}$ and $R'_t = \emptyset$ for all $t \in \mathcal{M}^{\mathcal{OD}}$.
   Let Y be a $\mathcal{C}L$-saturation of X and let $H(\tau) = \mathsf{compact}(Y)$.
   Mark τ 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 $U = \mathsf{trans}(H(w), t) \cup \{\varphi\}$ be the result of applying rule (trans) to $H(w)$, let Y be a $\mathcal{C}L$-saturation of U, and let $Z = \mathsf{compact}(Y)$.
      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 $t \in \mathcal{EI}(L)$ such that $R'_t(v,w)$ does not hold for any v:
      i. Let U be a $\mathcal{C}L$-saturation of $\mathsf{trans}_4(H(w), \langle t \rangle \top)$.
      ii. While there exist $\langle t \rangle \varphi \in U$ and a $\mathcal{C}L$-saturation V of $\mathsf{trans}_4(U, \langle t \rangle \varphi)$ such that $\mathsf{core}_5(U,t) \subset \mathsf{core}_5(V,t)$, let $U = V$.
      iii. Let $\mathsf{core}^*_5(w,t) = \mathsf{core}_5(U,t)$.
      iv. For every $\langle t \rangle \varphi \in \mathsf{core}^*_5(w,t)$:
         – Let Y be a $\mathcal{C}L$-saturation of $\mathsf{trans}_4(\mathsf{core}^*_5(w,t), \langle t \rangle \varphi)$.
         – Let $Z = \mathsf{compact}(Y)$.
         – 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 $\mathsf{core}^*_5(w,t) = \mathsf{core}_5(H(w),t)$.
   (d) Mark w as resolved.
3. Let $(R_t)_{t \in \mathcal{M}^{\mathcal{OD}}}$ be the least extension of $(R'_t)_{t \in \mathcal{M}^{\mathcal{OD}}}$ for $t \in \mathcal{M}^{\mathcal{OD}}$ such that $\langle W, \tau, (R_t)_{t \in \mathcal{M}^{\mathcal{OD}}} \rangle$ is an L-frame (note that the seriality conditions are cared by the tableau rule (D) and need not to be considered here).

This algorithm always terminates: eventually, for every w, either w contains no $\langle t \rangle$-formulae, or there exists an ancestor with $H(u) = Z$ at Step 2(a)ii because all $\mathcal{C}L$-saturated sets are drawn from the finite and fixed set $X^*_{\mathcal{C}L}$.

4.3 Completeness Proof

Lemma 7. *The following assertions are invariants during execution of Step 3 of Algorithm 1 (when $(R'_t)_{t \in \mathcal{M}^{\mathcal{OD}}}$ are extended to $(R_t)_{t \in \mathcal{M}^{\mathcal{OD}}}$).*

1. *If $R_t(w,u)$ holds and $t \in \mathcal{EI}(L)$ then $\mathsf{core}^*_5(w,t) = \mathsf{core}^*_5(u,t) = \mathsf{core}_5(H(u),t)$.*
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 $\mathcal{L}(A_s, \delta_s(Q,t)) \subseteq \mathcal{L}(A_s, Q')$.*

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 $Q = \widetilde{\delta}_s(I_s, \alpha_1) \cup \ldots \cup \widetilde{\delta}_s(I_s, \alpha_k)$. By the computation of

$\mathsf{core}_5^*(w,t)$, we have $[A_s,\widetilde{\delta}_s(I_s,\alpha_i t)]\varphi \in \mathsf{core}_5^*(w,t)$, for $1 \leq i \leq k$. Hence $[A_s,\widetilde{\delta}_s(I_s,\alpha_i t)]\varphi \in \mathsf{core}_5(H(u),t)$ for every $1 \leq i \leq k$. It follows that there exists $[A_s,Q']\varphi \in H(u)$ such that $\delta_s(Q,t) \subseteq Q'$, 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 $t \in \mathcal{M}_{\mathcal{OD}}$ to R_t.

Consider the base case, when $R'_t(w,u)$ holds. For the first assertion, assume that $t \in \mathcal{EI}(L)$. Hence u must have been created from w via Step 2b. We have that $\mathsf{core}_5^*(w,t) = \mathsf{core}_5(H(u),t)$, because $\mathsf{core}_5^*(w,t)$ is $\mathsf{core}_5(t)$-saturated and u is created from w via R'_t using $\mathsf{core}_5^*(w,t)$ as the content of w. When u is resolved, we have that $\mathsf{core}_5^*(u,t) = \mathsf{core}_5(H(u),t)$ due to Step 2c. Hence the first assertion holds. The second assertion clearly holds for the case $t \notin \mathcal{EI}(L)$.

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, $\mathsf{core}_5^*(w,t) = \mathsf{core}_5^*(v,t)$ and $\mathsf{core}_5^*(v,t) = \mathsf{core}_5^*(u,t) = \mathsf{core}_5(H(u),t)$, which imply the first assertion. If $R_t(w,u)$ is created from $R_t(v,w)$ and $R_t(v,u)$ then, by the inductive assumption, $\mathsf{core}_5^*(v,t) = \mathsf{core}_5^*(w,t)$ and $\mathsf{core}_5^*(v,t) = \mathsf{core}_5^*(u,t) = \mathsf{core}_5(H(u),t)$, which imply the first assertion.

Consider the inductive step for the second assertion and the case when $t \notin \mathcal{EI}(L)$. 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 2(1), $\mathcal{L}(A_s,\delta_s(Q,t)) \subseteq \mathcal{L}(A_s,\widetilde{\delta}_s(Q,s_1\ldots s_k))$. 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 2(3), $\mathcal{L}(A_s,\widetilde{\delta}_s(Q,s_1\ldots s_i)) \subseteq \mathcal{L}(A_s,Q_i)$ since $\mathcal{L}(A_s,\widetilde{\delta}_s(Q,s_1\ldots s_{i-1})) \subseteq \mathcal{L}(A_s,Q_{i-1})$ and $\widetilde{\delta}_s(Q,s_1\ldots s_i) = \delta_s(\widetilde{\delta}_s(Q,s_1\ldots s_{i-1}),s_i)$. Hence $\mathcal{L}(A_s,\widetilde{\delta}_s(Q,s_1\ldots s_k)) \subseteq \mathcal{L}(A_s,Q_k)$. It follows that $\mathcal{L}(A_s,\delta_s(Q,t)) \subseteq \mathcal{L}(A_s,Q_k)$. Choose $Q' = Q_k$.

Lemma 8. *Let X be a finite $\mathcal{C}L$-consistent set of formulae in the primitive language and $M = \langle W,\tau,(R_t)_{t\in\mathcal{M}_{\mathcal{OD}}},H\rangle$ 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 $\mathcal{C}L$-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 $\langle t\rangle\varphi \in H(w)$ 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 7, there exists $[A_t,Q']\varphi \in H(u)$ such that $\mathcal{L}(A_t,\delta_t(I_t,t)) \subseteq \mathcal{L}(A_t,Q')$. Since $t \in \mathcal{L}(A_t)$, we have that $\varepsilon \in \mathcal{L}(A_t,\delta_t(I_t,t))$. Hence $\varepsilon \in \mathcal{L}(A_t,Q')$, 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 $[t]\varphi \in \mathsf{core}_5^*(w,t)$. Since $R_t(w,u)$ holds, there exists v such that $R'_t(v,u)$ holds. By Lemma 7, $\mathsf{core}_5^*(w,t) = \mathsf{core}_5^*(u,t) = \mathsf{core}_5^*(v,t)$. Hence $[t]\varphi \in \mathsf{core}_5^*(v,t)$. Since $R'_t(v,u)$ holds, it follows that $\varphi \in H(u)$.

We now prove the second assertion. Suppose $\langle t\rangle\varphi \in H(w)$ 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 $\langle t \rangle \varphi \in \mathsf{core}_5^*(w,t)$. Alternatively, suppose $R'_t(v,w)$ does hold for some v when w is resolved (at Step 2c). Since $\langle t \rangle \varphi \in H(w)$, we have $\langle t \rangle \varphi \in \mathsf{core}_5(H(w),t) = \mathsf{core}_5^*(v,t)$ by Lemma 7. 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 $\langle t \rangle \varphi \in \mathsf{core}_5^*(v,t)$, 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 3 and 8.

Theorem 1. *The calculus $\mathcal{C}L$ for $\mathcal{BR}eg$ logics is sound and complete.*

We use Algorithm 1 only to prove completeness of the calculus $\mathcal{C}L$ for a $\mathcal{BR}eg$ logic L. It assumes that the input set X is $\mathcal{C}L$-consistent and is inefficient due to the naive computation of saturations and the limited caching. In the Appendix, we present Algorithm 2 with global caching for checking $\mathcal{C}L$-consistency of formula sets. Since the calculus $\mathcal{C}L$ is sound and complete, $\mathcal{C}L$-consistency coincides with L-satisfiability. Algorithm 2 explores the search space by building an and-or graph using the tableau rules of $\mathcal{C}L$. The content (label) of a node in the graph is a formula set in the compact form. Global caching means that for each possible content, at most one node with that content in the search space is expanded, and such an expansion is done at most once for that content. Global caching is one of the most useful optimisations for tableau decision procedures for modal logics. Due to global caching and the compact form of nodes, Algorithm 2 has the optimal EXPTime complexity.

5 Conclusions

We have given an analytic cut-free tableau calculus for a large class $\mathcal{BR}eg$ of epistemic logics for reasoning about agent beliefs. As demonstrated for the wise men puzzle and its modified version, $\mathcal{BR}eg$ logics are very useful for reasoning about mutual beliefs of agents. The class $\mathcal{BR}eg$ enssentially extends the class of regular grammar logics by allowing axioms (D) and (5) which are useful and sometimes necessary for practical applications. Our tableau calculus for $\mathcal{BR}eg$ seems a simple extension of our tableau calculus for regular grammar logics [10] using standard tableau rules to deal with axioms (D) and (5). But, note that non-trivial complications lie in the use of finite automata specifying the extended grammar $eRG(L)$ instead of $RG(L)$. Our completeness proof for $\mathcal{BR}eg$ is also more sophisticated than for the case of regular grammar logics. Applying global caching to our calculus, we obtain the first optimal (EXPTime) tableau decision procedure for $\mathcal{BR}eg$, which does not use cut rules. Furthermore, it is easy to show that most of the well-known optimisation techniques for tableau decision procedures (as discussed in [11]) are applicable to this decision procedure.

References

1. H. Aldewereld, W. van der Hoek, and J.-J.Ch. Meyer. Rational teams: Logical aspects of multi-agent systems. *Fundamenta Informaticae*, 63(2–3):159–183, 2004.
2. F. Baader. Augmenting concept languages by transitive closure of roles: An alternative to terminological cycles. In *Proceedings of IJCAI'91*, pages 446–451, 1991.

3. M. Baldoni. Normal multimodal logics with interaction axioms. In D. Basin et al, editors, *Labelled Deduction*, pages 33–57. Kluwer Academic Publishers, 2000.
4. M. Baldoni, L. Giordano, and A. Martelli. A tableau for multimodal logics and some (un)decidability results. In *TABLEAUX'1998, LNCS 1397:44-59*, 1998.
5. G. De Giacomo and F. Massacci. Combining deduction and model checking into tableaux and algorithms for Converse-PDL. *Inf. and Comp.*, 117-137:87–138, 2000.
6. S. Demri. The complexity of regularity in grammar logics and related modal logics. *Journal of Logic and Computation*, 11(6):933–960, 2001.
7. 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.
8. R. Goré. Tableau methods for modal and temporal logics. In D'Agostino et al, editor, *Handbook of Tableau Methods*, pages 297–396. Kluwer, 1999.
9. 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.
10. R. Goré and L.A. Nguyen. EXPTIME tableaux with global caching for description logics with transitive roles, inverse roles and role hierarchies. Accepted for TABLEAUX, 2007.
11. R. Goré and L.A. Nguyen. Optimised EXPTIME tableaux for $\mathcal{ALC}$ using sound global caching, propagation and cutoffs. Submitted to AIJ. Available at `http://www.mimuw.edu.pl/~nguyen/GoreNguyenAIJ.pdf`, 2007.
12. J.Y. Halpern and Y. Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence*, 54:319–379, 1992.
13. D. Harel, D. Kozen, and J. Tiuryn. *Dynamic Logic*. MIT Press, 2000.
14. I. Horrocks and U. Sattler. Decidability of SHIQ with complex role inclusion axioms. *Artif. Intell.*, 160(1-2):79–104, 2004.
15. K. Konolige. Belief and incompleteness. Technical Report 319, SRI Inter., 1984.
16. A. Mateescu and A. Salomaa. Formal languages: an introduction and a synopsis. In *Handbook of Formal Languages - Volume 1*, pages 1–40. Springer, 1997.
17. J. McCarthy. First order theories of individual concepts and propositions. *Machine Intelligence*, 9:120–147, 1979.
18. L.A. Nguyen. Analytic tableau systems and interpolation for the modal logics KB, KDB, K5, KD5. *Studia Logica*, 69(1):41–57, 2001.
19. L.A. Nguyen. Analytic tableau systems for propositional bimodal logics of knowledge and belief. In *TABLEAUX 2002, LNAI 2381:206-220*. Springer, 2002.
20. L.A. Nguyen. Multimodal logic programming. *Theoretical Comp. Sci.*, 360:247–288, 2006.
21. L.A. Nguyen. On the deterministic Horn fragment of test-free PDL. In *Advances in Modal Logic - Volume 6*, pages 373–392. King's College Publications, 2006.
22. L.A. Nguyen. Reasoning about epistemic states of agents by modal logic programming. In F. Toni and P. Torroni, editors, *Proc. of CLIMA VI, LNAI 3900*, pages 37–56. Springer, 2006.
23. W. Rautenberg. Modal tableau calculi and interpolation. *JPL*, 12:403–423, 1983.
24. J. van Benthem. Correspondence theory. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic, Vol II*, pages 167–247. Reidel, Dordrecht, 1984.

Appendix: An EXPTime Decision Procedure with Global Caching for $\mathcal{BR}eg$ Logics

Algorithm 2

Input: a finite set X of primitive language formulae and an $\mathcal{BR}eg$ logic L with finite automata $(A_t)_{t\in\mathcal{MOD}\setminus\mathcal{EI}(L)}$ specifying the extended grammar $eRG(L)$
Output: an and-or graph $G=\langle V,E\rangle$ with $\tau\in V$ as the initial node such that $\tau.status=$ `cons` iff X is $\mathcal{C}L$-consistent
Remark: We use "rule" to refer to a $\mathcal{C}L$-tableau rule.

1. create a new node τ with $\tau.content := X$ and $\tau.status :=$ `unexpanded`; let $V := \{\tau\}$ and $E := \emptyset$;
2. while $\tau.status \notin \{$`cons`, `incons`$\}$ and we can choose an unexpanded node $v \in V$ do:
   (a) $\mathcal{D} := \emptyset$;
   (b) if no rule is applicable to *v.content* then *v.status* := `cons`
   (c) else if the rule $(\bot)$ is applicable to *v.content* then *v.status* := `incons`
   (d) else if some static rule with only one denominator is applicable to *v.content* giving denominator Y then *v.kind* := `and-node`, $\mathcal{D} := \{Y\}$
   (e) else if the rule $(\vee)$ is applicable to *v.content* giving denominators Y_1 and Y_2 (both different from *v.content*) then *v.kind* := `or-node`, $\mathcal{D} := \{Y_1, Y_2\}$
   (f) else
      i. *v.kind* := `and-node`,
      ii. for every transitional rule applicable to *v.content* and for every possible application of the rule to *v.content* giving denominator Y, add Y to $\mathcal{D}$;
   (g) for every denominator $Y \in \mathcal{D}$ do
      i. let $Z = \mathsf{compact}(Y)$,
      ii. if some $w \in V$ has *w.content* $= Z$ then add edge (v,w) to E
      iii. else let w be a new node, set *w.content* $:= Z$, *w.status* := `unexpanded`, add w to V, and add edge (v,w) to E;
   (h) if (*v.kind* = `or-node` and one of the successors of v has status `cons`) or (*v.kind* = `and-node` and all the successors of v have status `cons`) then *v.status* := `cons`, *propagate*(G,v)
   (i) else if (*v.kind* = `and-node` and one of the successors of v has status `incons`) or (*v.kind* = `or-node` and all the successors of v have status `incons`) then *v.status* := `incons`, *propagate*(G,v)
   (j) else *v.status* := `expanded`;
3. if $\tau.status \notin \{$`cons`, `incons`$\}$ then for every node $v \in V$ with *v.status* $\neq$ `incons`, set *v.status* := `cons`;

Fig. 2. Checking $\mathcal{C}L$-Consistency Using Global Caching

In this section L denotes an $\mathcal{BR}eg$ logic. In Figure 2 we give an algorithm for checking $\mathcal{C}L$-consistency which creates an and-or graph using the tableau rules of $\mathcal{C}L$ and global caching. A node in the constructed graph is a record with three attributes:

content: the formula set carried by the node

Procedure *propagate*(G,v)
Parameters: an and-or graph $G = \langle V,E \rangle$ and $v \in V$ with $v.status \in \{\texttt{cons}, \texttt{incons}\}$
Returns: a modified and-or graph $G = \langle V,E \rangle$

1. $queue := \{v\}$;
2. while *queue* is not empty do
3. (a) extract x from *queue*;
   (b) for every $u \in V$ with $(u,x) \in E$ and $u.status = \texttt{expanded}$ do
      i. if ($u.kind = \texttt{or-node}$ and one of the successors of u has status `cons`)
         or ($u.kind = \texttt{and-node}$ and all the successors of u have status `cons`) then
            $u.status := \texttt{cons}$, $queue := queue \cup \{u\}$
      ii. else if ($u.kind = \texttt{and-node}$ and one of the successors of u has status `incons`)
         or ($u.kind = \texttt{or-node}$ and all the successors of u have status `incons`) then
            $u.status := \texttt{incons}$, $queue := queue \cup \{u\}$;

Fig. 3. Propagating Consistency and Inconsistency Through an And-Or Graph

status: $\{\texttt{unexpanded}, \texttt{expanded}, \texttt{cons}, \texttt{incons}\}$
kind: $\{\texttt{and-node}, \texttt{or-node}\}$

To check whether a given finite formula set X is $\mathcal{C}L$-consistent, the initial node τ has content X and status `unexpanded`. The main while-loop continues processing nodes until the status of τ is determined to be in $\{\texttt{cons}, \texttt{incons}\}$, or until every node is expanded, whichever happens first.

The algorithm gives a preference to the rule $(\bot)$, then any one of the static unary rules, then the static binary rule $(\vee)$. If none of these are applicable, then it applies the transitional rules simultaneously.

When a rule is applied, the algorithm categorises the numerator as either an `or-node` or an `and-node` with an `or-node` being inconsistent if every child is inconsistent and an `and-node` being inconsistent if at least one child is inconsistent.

The main difference with traditional methods appears at Step 2g: here, for every denominator, we first check whether an already existing node can act as a proxy for that denominator. If so, then we do not create that denominator, but merely insert an edge from the numerator to the existing proxy.

If these steps cannot determine the status of v as `cons` or `incons`, then its status is set to `expanded`. But if these steps do determine the status of a node v to be `cons` or `incons`, this information is itself propagated to the predecessors of v in the and-or graph G via the routine *propagate*(G,v), explained shortly.

The main loop ends when the status of the initial node τ becomes `cons` or `incons` or all nodes of the graph have been expanded. In the latter case, all nodes with status $\neq$ `incons` are given status `cons` (effectively giving the status *open* to tableau branches which loop).

The procedure *propagate* used in the above algorithm is specified in Figure 3. As parameters, it accepts an and-or graph G and a node v with (irrevocable) status `cons` or `incons`. The purpose is to propagate the status of v through the and-or graph and alter G to reflect the new information.

Initially, the queue contains only v. While the queue is not empty: a node x is extracted; the status of x is propagated to each predecessor u of x in an appropriate way; and if the status of u becomes (irrevocably) `cons` or `incons` then u is inserted into the queue for further propagation.

This construction thus uses both caching and propagation techniques.

Lemma 9. *It is an invariant of Algorithm 2 that for every $v \in V$:*

1. *if v.status* $=$ `incons` *then*
   - *the* $(\bot)$*-rule of* $\mathcal{C}L$ *is applicable to v.content,*
   - *or v.kind* $=$ `and-node` *and there exists* $(v,w) \in E$ *such that* $w \neq v$ *and w.status* $=$ `incons`*,*
   - *or v.kind* $=$ `or-node` *and for every* $(v,w) \in E$*, w.status* $=$ `incons`*;*
2. *if v.status* $=$ `cons` *then*
   - *no rule of* $\mathcal{C}L$ *is applicable to v.content,*
   - *or v.kind* $=$ `or-node` *and there exists* $(v,w) \in E$ *with w.status* $=$ `cons`*,*
   - *or v.kind* $=$ `and-node` *and for every* $(v,w) \in E$*, w.status* $=$ `cons`*.*

(Since a static rule is applied to X only when X is not closed w.r.t. the rule, if v.kind $=$ `or-node` *and* $(v,w) \in E$ *then* $w \neq v$ *since w.content* $\neq$ *v.content.)*

Proof. Lemma 9(1) clearly holds since these are the only three ways for a node to get status `incons`. For Lemma 9(2) there is the possibility that the node gets status `cons` via Step 3 of Algorithm 2.

For a contradiction, assume that *v.status* becomes `cons` because of Step 3 of Algorithm 2 and that all three clauses of the "then" part of Lemma 9(2) fail:

1. First, the rule assumed to be applicable to *v.content* cannot be the $(\bot)$-rule as this would have put *v.status* $=$ `incons`, contradicting our assumption that *v.status* $=$ `cons`. Hence *v.kind* $=$ `or-node` or *v.kind* $=$ `and-node` after this rule application.
2. Second, if *v.kind* $=$ `or-node` then v must have two successors created by the rule $(\vee)$. If none of the successors has status `cons` then they must all have status `incons`. But Algorithm 2 and procedure *propagate* always ensure that `incons` is propagated whenever it is found. As soon as the `incons` status of the lattest of the children is found, the ensuing call to *propagate* would have ensured that *v.status* $=$ `incons`, contradicting our assumption that *v.status* $=$ `cons`.
3. Third, if *v.kind* $=$ `and-node` then v has at least one successor w (say) with $(v,w) \in E$. If *w.status* $\neq$ `cons`, then we must have *w.status* $=$ `incons`. Again, when w gets status `incons`, procedure *propagate* would ensure that *v.status* $=$ `incons` too, contradicting our assumption that *v.status* $=$ `cons`.

Lemma 10. *Let $G = \langle V,E \rangle$ be the graph constructed by Algorithm 2 for X using $\mathcal{C}L$. If* τ*.status* $=$ `incons` *then X is $\mathcal{C}L$-inconsistent.*

Proof. Using Lemma 9, we can construct a closed $\mathcal{C}L$-tableau for X by induction on the way a node depends on its successors and by copying nodes so that the resulting structure is a (tree) tableau rather than a graph.

Let $G = \langle V, E \rangle$ be the graph constructed by Algorithm 2 for X using $\mathcal{C}L$. For $v \in V$ with $v.status = \texttt{cons}$, we say that $v_0 = v, v_1, \ldots, v_k$ with $k \geq 0$ is a *saturation path of v in G* if for each $1 \leq i \leq k$, we have $v_i.status = \texttt{cons}$, the edge $E(v_{i-1}, v_i)$ was created by an application of a static rule, and $v_k.content$ is closed w.r.t. the static rules. Observe that if $v_0, \ldots, v_k$ is a saturation path of v_0 in G then $v_0.content \subseteq \ldots \subseteq v_k.content$. By Lemma 9, if $v.status = \texttt{cons}$ then there exists a saturation path of v in G.

Lemma 11. *Let $G = \langle V, E \rangle$ be the graph constructed by Algorithm 2 for X using $\mathcal{C}L$. If $\tau.status = \texttt{cons}$ then every $\mathcal{C}L$-tableau for X is open.*

Proof. Let T be an arbitrary $\mathcal{C}L$-tableau for X. We maintain a *current node cn* of T that will follow edges of T to pin-point an open branch of T. Initially we set *cn* to be the root of T. We also keep a (finite) saturation path σ of the form $\sigma_0, \ldots, \sigma_k$ for some $\sigma_0 \in V$ and call σ the *current saturation path in G*. At the beginning, set $\sigma_0 := \tau$ and let σ be a saturation path for σ_0 in G: such a saturation path exists since $\tau.status = \texttt{cons}$. We maintain the invariant $cn.content \subseteq \sigma_k.content$, where $cn.content$ is the set carried by *cn*.

Remark 1. By the definition of saturation path, $\sigma_k.status = \texttt{cons}$. The invariant thus implies that the $(\bot)$-rule is not applicable to *cn*.

Clearly, the invariant holds at the beginning since $\sigma_0 = \tau$ and $\tau.content = cn.content$ and $\sigma_0.content \subseteq \sigma_k.content$. Depending upon the rule applied to *cn* in the tableau T, we maintain the invariant by changing the value of the current node *cn* of T and possibly also the current saturation path σ in G:

1. Case the tableau rule applied to *cn* is a static rule. Since $cn.content \subseteq \sigma_k.content$ and $\sigma_k.content$ is closed w.r.t. the static rules, *cn* has a successor u in T with $u.content \subseteq \sigma_k.content$. By setting $cn := u$, the invariant is maintained without changing σ.
2. Case the tableau rule applied to *cn* is a transitional rule and the successor is $u \in T$. By the invariant, the rule can be applied to $\sigma_k.content$ in the same way, creating a successor node $w \in V$ with $w.content \supseteq u.content$. Moreover, σ_k is an and-node with $\sigma_k.status = \texttt{cons}$, hence $w.status \neq \texttt{incons}$, meaning that $w.status = \texttt{cons}$. Setting $cn := u$ and setting σ to be a saturation path of w in G maintains the invariant.

By Remark 1, the branch formed by the instances of *cn* is an open branch of T.

Theorem 2. *Let L be an $\mathcal{BR}eg$ logic whose extended grammar is specified by finite automata $(A_t)_{t \in \mathcal{M}_{OD}}$, X a finite set of primitive language formulae, and $G = \langle V, E \rangle$ the graph constructed by Algorithm 2 for X using $\mathcal{C}L$, with $\tau \in V$ as the initial node. Then X is $\mathcal{C}L$-consistent iff $\tau.status = \texttt{cons}$.*

This theorem immediately follows from Lemmas 10 and 11.

Corollary 1. *Algorithm 2 is an EXPTime decision procedure for $\mathcal{BR}eg$ logics.*

Proof. Let L be an $\mathcal{BR}eg$ logic and X a finite formula set in the primitive language. Since $\mathcal{C}L$ is sound and complete (Theorem 1), X is L-satisfiable iff X is $\mathcal{C}L$-consistent, and iff the execution of Algorithm 2 for X and L returns a graph with $\tau.status = \texttt{cons}$ (by Theorem 2).

Let n be the sum of the sizes of the formulae in X and the sizes of the automata specifying $eRG(L)$. Assume that $n > |\mathcal{M}_{\mathcal{O}\mathcal{D}}|$. 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 $3n$ i.e. $O(n)$ formulae. Hence there are at most $(2^{O(n)})^{O(n)} = 2^{O(n^2)}$ different node contents. Due to global caching, each node in the constructed and-or graph has a unique content, so the graph has at most $2^{O(n^2)}$ nodes.

Every $v \in V$ is expanded (by Steps (2a)–(2j)) only once and such a task takes $2^{O(n^2)}$ time units without counting the execution time of the procedure *propagate*. When *v.status* becomes `cons` or `incons`, the procedure *propagate* executes $2^{O(n^2)}$ basic steps directly involved with v. Hence the total time of the executions of *propagate* is of rank $2^{O(n^2)}$. The time complexity of Algorithm 2 is therefore of rank $2^{O(n^2)}$.