2008 Sixth IEEE International Conference on Software Engineering and Formal Methods

Tableau-based decision procedure for the multi-agent epistemic logic with operators of common and distributed knowledge

Valentin Goranko
University of the Witwatersrand
School of Mathematics
WITS 2050, Johannesburg, South Africa
goranko@maths.wits.ac.za

Dmitry Shkatov
University of the Witwatersrand
School of Computer Science
WITS 2050, Johannesburg, South Africa
dmitry@cs.wits.ac.za

Abstract

*We develop an incremental-tableau-based decision procedure for the multi-agent epistemic logic **MAEL**(CD) (aka $S5_n(CD)$), whose language contains operators of individual knowledge for a finite set Σ of agents, as well as operators of distributed and common knowledge among all agents in Σ. Our tableau procedure works in (deterministic) exponential time, thus establishing an upper bound for **MAEL**(CD)-satisfiability that matches the (implicit) lower-bound known from earlier results, which implies* $\mathbf{ExpTime}$*-completeness of **MAEL**(CD)-satisfiability. Therefore, our procedure provides a complexity-optimal algorithm for checking **MAEL**(CD)-satisfiability, which, however, in most cases is much more efficient. We prove soundness and completeness of the procedure, and illustrate it with an example.*

1. Introduction

Over the last two decades, multi-agent epistemic logics ([2, 8]) have played a significant role in computer science and artificial intelligence. The main application seems to have been to design, specification, and verification of distributed protocols ([6]), but a plethora of other applications are described in, among others, [3], [2] and [8].

Languages of multi-agent epistemic logics considered in the literature contain various repertoires of modal operators. In the present paper, we consider the "full" multi-agent epistemic logic, which we call **MAEL**(CD), whose language contains operators of individual knowledge for a non-empty, finite set Σ of agents as well as operators of common (C) and distributed (D) knowledge among all agents in Σ. (Since all modal operators of **MAEL**(CD) are $\mathbf{S5}$-modalities, the logic is also referred to in the literature as $S5_n(CD)$). To be used for such tasks as designing protocols conforming to a given specification, **MAEL**(CD), needs to be equipped with an algorithm checking for **MAEL**(CD)-satisfiability. The first step in that direction was taken in [10], where the decidability of **MAEL**(CD) has been established by showing that it has a finite model property. This result was proved in [10] via filtration; therefore, the decision procedure suggested by that argument is based on an essentially brute-force enumeration of all finite models for **MAEL**(CD), which suggest a satisfiability-checking algorithm that is theoretically important, but of limited practical value. Our tableau procedure has, in comparison, the following advantages:

1. It establishes a (deterministic) $\mathbf{ExpTime}$ upper-bound for **MAEL**(CD)-satisfiability, which matches the lower-bound that follows from the results of [7].
2. It provides an algorithm for checking **MAEL**(CD)-satisfiability that is not only provably complexity-optimal, but which in the vast majority of cases requires much less resources than what is predicted by the worst-case upper bound. This is one of the hallmarks of incremental tableaux ([11]) as opposed to the top-down tableaux in the style of [1], which *always* require the amount of resources predicted by the worst-case complexity estimate. Top-down tableaux for the fragment of **MAEL**(CD) not containing the operator of distributed knowledge have been presented in [7].

The type of incremental tableau developed herein originates in [11]; tableaux in a similar style were recently developed for the multi-agent logic **ATL** and some of its variations in [5]. Thus, the present paper continues the enterprize of designing complexity-optimal decision procedures for logics used in design, specification and verification of multi-agent systems ([2, 12]). The particular style of the tableaux presented here is meant to be compatible with the tableaux from [5], so that we can in the future build tableaux for more sophisticated logics for multi-agent systems.

978-0-7695-3437-4/08 $25.00 © 2008 IEEE
DOI 10.1109/SEFM.2008.27

The main reason for the restriction of the distributed and common knowledge operators only to be (implicitly) parameterized by the whole set of agents referred to in the language, adopted in this paper, is to be able to present the main ideas and features of the tableaux in sufficient detail, while avoiding some additional technical complications arising in the case of several such operators, each one associated with a non-empty subset of the set of all agents. This, more complicated, case will be treated in a follow-up paper.

2. Syntax and semantics of MAEL(CD)

2.1. Syntax

The language $\mathcal{L}$ of **MAEL**(CD) contains a (possibly, countably-infinite) set $\mathtt{AP}$ of *atomic propositions*, typically denoted by $p, q, r, \ldots$; a finite, non-empty set Σ of (names of) *agents*, typically denoted by $a, b \ldots$; a sufficient repertoire of the Boolean connectives; and the modal operators $\mathbf{K}_a$ ("the agent a knows that ..."), $\mathbf{D}$ ("it is distributed knowledge among Σ that ...") and $\mathbf{C}$ ("it is common knowledge among Σ that ..."). Thus, the formulae of $\mathcal{L}$ are defined as follows:

$$\varphi := p \mid \neg(\varphi) \mid (\varphi_1 \wedge \varphi_2) \mid \mathbf{K}_a(\varphi) \mid \mathbf{D}(\varphi) \mid \mathbf{C}(\varphi),$$

where p ranges over $\mathtt{AP}$ and a ranges over Σ. The other boolean connectives can be defined in the usual way. We omit parentheses in formulae whenever it does not result in ambiguity. We denote arbitrary formulae of $\mathcal{L}$ by $\varphi, \psi, \chi, \ldots$ (possibly with decorations). We write $\varphi \in \mathcal{L}$ to mean that φ is a formula of $\mathcal{L}$. Formulae of the form $\neg\mathbf{C}\varphi$ are called *eventualities*.

2.2. Semantics

Formulae of $\mathcal{L}$ are interpreted over multi-agent epistemic models, based on multi-agent epistemic frames. We will also need a more general notion of multi-agent epistemic structure.

Definition 2.1 *A* multi-agent epistemic structure *(MAES, for short) is a tuple $\mathfrak{S} = (\Sigma, S, \{\mathcal{R}_a\}_{a\in\Sigma}, \mathcal{R}_D, \mathcal{R}_C)$, where*

1. *Σ is a finite, non-empty set of agents;*
2. *$S \neq \emptyset$ is a set of* states*;*
3. *$\mathcal{R}_D$ and $\mathcal{R}_a$, for each $a \in \Sigma$, are binary relations on S;*
4. *$\mathcal{R}_C$ is the transitive closure of $\mathcal{R}_D \cup \bigcup_{a\in\Sigma}\mathcal{R}_a$.*

Definition 2.2 *A* multi-agent epistemic frame *(MAEF, for short) is a MAES $\mathfrak{F} = (\Sigma, S, \{\mathcal{R}_a\}_{a\in\Sigma}, \mathcal{R}_D, \mathcal{R}_C)$, where*

(a) *$\mathcal{R}_D$ and $\mathcal{R}_a$, for every $a \in \Sigma$, are equivalence relations on S;*

(b) *$\mathcal{R}_D = \bigcap_{a\in\Sigma}\mathcal{R}_a$.*

If condition (b) above is replaced with

(b') *$\mathcal{R}_D \subseteq \bigcap_{a\in\Sigma}\mathcal{R}_a$,*

then $\mathfrak{F}$ is a multi-agent epistemic pseudo-frame.

Notice that in (pseudo-)frames condition 4 of definition 2.1 is equivalent to the requirement that $\mathcal{R}_C$ is the transitive closure of $\bigcup_{a\in\Sigma}\mathcal{R}_a$. Also notice that, as in any MAEF each $\mathcal{R}_a$ is an equivalence relation, $\mathcal{R}_C$ is also an equivalence relation.

Definition 2.3 *A* multi-agent epistemic model *(MAEM, for short) is a tuple $\mathcal{M} = (\mathfrak{F}, \mathtt{AP}, L)$, where*

(i) $\mathfrak{F}$ is a MAEF;
(ii) $\mathtt{AP}$ is a (possibly, infinite) set of atomic propositions;
(iii) $L : S \mapsto \mathcal{P}(\mathtt{AP})$, is a labeling function*, where $L(s)$ is the set of all atomic propositions that are declared true at s.*

If condition (i) above is replaced by the requirement that $\mathfrak{F}$ is a multi-agent epistemic pseudo-frame, then $\mathcal{M}$ is a multi-agent epistemic pseudo-model *(pseudo-MAEM).*

The satisfaction relation between (pseudo-)MAEMs and formulae is defined in the standard way. In particular,

- $\mathcal{M}, s \Vdash \mathbf{K}_a\varphi$ iff $(s,t) \in \mathcal{R}_a$ implies $\mathcal{M}, t \Vdash \varphi$;
- $\mathcal{M}, s \Vdash \mathbf{D}\varphi$ iff $(s,t) \in \mathcal{R}_D$ implies $\mathcal{M}, t \Vdash \varphi$;
- $\mathcal{M}, s \Vdash \mathbf{C}\varphi$ iff $(s,t) \in \mathcal{R}_C$ implies $\mathcal{M}, t \Vdash \varphi$.

The truth condition for the operator $\mathbf{C}$ can be paraphrased in terms of reachability. Let $\mathfrak{F}$ be a (pseudo-)frame with state space S and let $s, t \in S$. We say that t *is reachable from* s if there exists a sequence $s = s_0, s_1, \ldots, s_{n-1}, s_n = t$ of elements of S such that, for every $0 \leq i < n$, there exists $a \in \Sigma$ such that $(s_i, s_{i+1}) \in R_a$. It is then easy to see that the following truth condition for $\mathbf{C}$ is equivalent in (pseudo-)MAEMs to the one given above:

- $\mathcal{M}, s \Vdash \mathbf{C}\varphi$ iff $\mathcal{M}, t \Vdash \varphi$ whenever t is reachable from s.

Notice that if $\Sigma = \{a\}$, then the formulae $\mathbf{K}_a\varphi \leftrightarrow \mathbf{D}\varphi$ and $\mathbf{K}_a\varphi \leftrightarrow \mathbf{C}\varphi$ are valid for all $\varphi \in \mathcal{L}$, so the one-agent case is trivialized. Thus, we assume throughout the remainder of the paper that Σ contains at least 2 agents.

Definition 2.4 (Satisfiability and validity)

- *Let $\varphi \in \mathcal{L}$ and $\mathcal{M}$ be a MAEM. We say that φ is* satisfiable *in $\mathcal{M}$ if $\mathcal{M}, s \Vdash \varphi$ holds for some $s \in \mathcal{M}$ and that φ is* valid *in $\mathcal{M}$ if $\mathcal{M}, s \Vdash \varphi$ holds for every $s \in \mathcal{M}$.*
- *Let $\varphi \in \mathcal{L}$ and M be a class of models. We say that φ is* satisfiable *in M if $\mathcal{M}, s \Vdash \varphi$ holds for some $\mathcal{M} \in \mathsf{M}$ and some $s \in \mathcal{M}$ and that φ is* valid *in M if $\mathcal{M}, s \Vdash \varphi$ holds for every $\mathcal{M} \in \mathsf{M}$ and every $s \in \mathcal{M}$.*

The goal of this paper is to develop a sound, complete, and complexity-optimal tableau-based decision procedure for testing satisfiability, and hence also validity, of formulas of $\mathcal{L}$ in the class of all MAEMs; in other words, the procedure tests for the belonging of formulae of $\mathcal{L}$ to the logic **MAEL**(CD), which is the logic of all such models.

3. Hintikka structures

The ultimate purpose of the tableau procedure we develop is to check if the input formula is satisfiable in a MAEM. However, the tableau attempts not to directly construct a MAEM for the input formula, but to build a more general kind of semantic structure, viz. a *Hintikka structure* (which are, therefore, used in proving completeness of our tableaux). The basic difference between models and Hintikka structures is that while models determine the truth of every formula of the language at every state, Hintikka structures only provide truth values of the formulae relevant to the evaluation of a fixed formula θ. Another important difference is that the accessibility relations in models must satisfy the explicitly stated conditions of definition 2.2, while in Hintikka structures we only impose conditions on the sets of formulas in the labels of the states, which correspond to the desirable conditions on the accessibility relations. Even though no conditions are implicitly imposed on the accessibility relations themselves, the labeling is done is such a way that every Hintikka structure generates, by a construction described in the proof of lemma 3.5, a MAEM in such a way that the "truth" of the formulas in the labels is preserved in the resultant model (whose relations satisfy all conditions of definition 2.2).

To define Hintikka structures, we need the following auxiliary notion, inspired by [7].

Definition 3.1 *A set $\Delta \subseteq \mathcal{L}$ is* fully expanded *if it satisfies the following conditions ($\mathsf{Sub}(\varphi)$ stands for the set of subformulae of the formula φ):*

- *if $\neg\neg\varphi \in \Delta$, then $\varphi \in \Delta$;*
- *if $\varphi \wedge \psi \in \Delta$, then $\varphi \in \Delta$ and $\psi \in \Delta$;*
- *if $\neg(\varphi \wedge \psi) \in \Delta$, then $\neg\varphi \in \Delta$ or $\neg\psi \in \Delta$;*
- *if $\mathbf{K}_a\varphi \in \Delta$, for some $a \in \Sigma$, then $\mathbf{D}\varphi \in \Delta$;*
- *if $\mathbf{D}\varphi \in \Delta$, then $\varphi \in \Delta$;*
- *if $\mathbf{C}\varphi \in \Delta$, then $\mathbf{K}_a(\varphi \wedge \mathbf{C}\varphi) \in \Delta$ for every $a \in \Sigma$;*
- *if $\neg\mathbf{C}\varphi \in \Delta$, then $\neg\mathbf{K}_a(\varphi \wedge \mathbf{C}\varphi) \in \Delta$ for some $a \in \Sigma$;*
- *if $\varphi \in \Delta$ and $\psi \in \mathsf{Sub}(\varphi)$ is of the form $\mathbf{K}_a\chi$ or $\mathbf{D}\chi$, then either $\psi \in \Delta$ or $\neg\psi \in \Delta$.*

Definition 3.2 *A* multi-agent epistemic Hintikka structure *(MAEHS for short) is a tuple $(\Sigma, S, \{\mathcal{R}_a\}_{a\in\Sigma}, \mathcal{R}_D, \mathcal{R}_C, H)$ such that*

- *$(\Sigma, S, \{\mathcal{R}_a\}_{a\in\Sigma}, \mathcal{R}_D, \mathcal{R}_C)$ is a MAES;*
- *H is a labeling of the elements of S with formulae of $\mathcal{L}$ that satisfies the following constraints:*

  H1 *if $\neg\varphi \in H(s)$, then $\varphi \notin H(s)$;*
  H2 *$H(s)$ is fully expanded, for every $s \in S$;*
  H3 *if $\mathbf{K}_a\varphi \in H(s)$ and $(s,t) \in \mathcal{R}_a$, then $\varphi \in H(t)$;*
  H4 *if $\neg\mathbf{K}_a\varphi \in H(s)$, then there exists $t \in S$ such that $(s,t) \in \mathcal{R}_a$ and $\neg\varphi \in H(t)$;*
  H5 *if $(s,t) \in \mathcal{R}_a$, then $\mathbf{K}_a\varphi \in H(s)$ iff $\mathbf{K}_a\varphi \in H(t)$;*
  H6 *if $\mathbf{D}\varphi \in H(s)$ and $(s,t) \in \mathcal{R}_D$, then $\varphi \in H(t)$;*
  H7 *if $\neg\mathbf{D}\varphi \in H(s)$, then there exists $t \in S$ such that $(s,t) \in \mathcal{R}_D$ and $\neg\varphi \in H(t)$;*
  H8 *if $(s,t) \in \mathcal{R}_D$, then $\mathbf{D}\varphi \in H(s)$ iff $\mathbf{D}\varphi \in H(t)$, and $\mathbf{K}_a\varphi \in H(s)$ iff $\mathbf{K}_a\varphi \in H(t)$, for every $a \in \Sigma$;*
  H9 *if $\neg\mathbf{C}\varphi \in H(s)$, then there exists $t \in S$ such that $(s,t) \in \mathcal{R}_C$ and $\neg\varphi \in H(t)$.*

Definition 3.3 *Let $\theta \in \mathcal{L}$ and $\mathcal{H}$ be a MAEHS with state space S. We say that $\mathcal{H}$ is a MAEHS for θ if $\theta \in H(s)$ for some $s \in S$.*

Now we will prove that $\theta \in \mathcal{L}$ is satisfiable in the class of all MAEMs iff there exists a MAEHS for θ. This will allow us to design our tableau procedure to test for the existence of a MAEHS, rather than a MAEM, for the input formula.

Given a MAEM $\mathcal{M}$ with a labeling function L, we define the *extended labeling function* $L^+ : S \mapsto \mathcal{P}(\mathcal{L})$ on $\mathcal{M}$ as follows: $L^+(s) = \{\, \varphi \mid \mathcal{M}, s \Vdash \varphi \,\}$. Then, the following is straightforward.

Lemma 3.4 *Let $\mathcal{M} = (\Sigma, S, \{\mathcal{R}_a\}_{a\in\Sigma}, \mathcal{R}_D, \mathcal{R}_C, L)$ be a MAEM satisfying θ and let L^+ be an extended labeling on $\mathcal{M}$. Then, $(\Sigma, S, \{\mathcal{R}_a\}_{a\in\Sigma}, \mathcal{R}_D, \mathcal{R}_C, L^+)$ is a MAEHS for θ.*

Next, we prove the opposite direction.

Lemma 3.5 *Let $\theta \in \mathcal{L}$ be such that there exists a MAEHS for θ. Then, θ satisfiable in a MAEM.*

Proof. Let $\theta \in \mathcal{L}$ and $\mathcal{H} = (\Sigma, S, \{\mathcal{R}_a\}_{a\in\Sigma}, \mathcal{R}_D, \mathcal{R}_C, H)$ be an MAEHS for θ. First, we define, using $\mathcal{H}$, a pseudo-MAEM $\mathcal{M}'$ satisfying θ; then, we turn $\mathcal{M}'$ into a MAEM satisfying θ.

$\mathcal{M}'$ is defined as follows. First, for every $a \in \Sigma$, let $\mathcal{R}'_a$ be the reflexive, symmetric, and transitive closure of $\mathcal{R}_a \cup \mathcal{R}_D$; let $\mathcal{R}'_D$ be the reflexive, symmetric, and transitive closure of $\mathcal{R}_D$; and let $\mathcal{R}'_C$ be the transitive closure of $\bigcup_{a\in\Sigma}\mathcal{R}'_a$. (Notice that $\mathcal{R}_C \subseteq \mathcal{R}'_C$.) Second, let $\mathtt{AP} = \{\, p \in H(t) \mid t \in S \text{ and } p \text{ is an atomic proposition} \,\}$. Finally, let $L(s) = H(s) \cap \mathtt{AP}$ for every $s \in S$. It is then straightforward

to check that $\mathcal{M}' = (\Sigma, S, \{\mathcal{R}'_a\}_{a\in\Sigma}, \mathcal{R}'_D, \mathcal{R}'_C, \mathtt{AP}, L)$ is a pseudo-MAEM (recall definition 2.3).

Next, we prove, by induction on the structure of $\chi \in \mathcal{L}$ that, for every $s \in S$ and every $\chi \in \mathcal{L}$, the following hold:

i) $\chi \in H(s)$ implies $\mathcal{M}', s \Vdash \chi$, and

ii) $\neg\chi \in H(s)$ implies $\mathcal{M}', s \Vdash \neg\chi$.

Let χ be some $p \in \mathtt{AP}$. Then, $p \in H(s)$ implies $p \in L(s)$ and, thus, $\mathcal{M}', s \Vdash p$; if, on the other hand, $\neg p \in H(s)$, then due to (H1), $p \notin H(s)$ and thus $p \notin L(s)$; hence, $\mathcal{M}', s \Vdash \neg p$.

Assume that the claim holds for all subformulae of χ; then, we have to prove that it holds for χ, as well.

Suppose that χ is $\neg\varphi$. If $\neg\varphi \in H(s)$, then the inductive hypothesis immediately gives us $\mathcal{M}', s \Vdash \neg\varphi$; if, on the other hand, $\neg\neg\varphi \in H(s)$, then by virtue of (H2), $\varphi \in H(s)$ and hence, by inductive hypothesis, $\mathcal{M}', s \Vdash \varphi$ and thus $\mathcal{M}', s \Vdash \neg\neg\varphi$.

The case of $\chi = \varphi \wedge \psi$ is straightforward, using (H2).

Suppose that χ is $\mathbf{K}_a\varphi$. Assume, first, that $\mathbf{K}_a\varphi \in H(s)$. In view of inductive hypothesis, it suffices to show that $(s,t) \in \mathcal{R}'_a$ implies $\varphi \in H(t)$. So, assume that $(s,t) \in \mathcal{R}'_a$. There are two cases to consider. If $s = t$, then the conclusion immediately follows from (H2). If, on the other hand, $s \neq t$, then there exists an undirected path from s to t along the relations $\mathcal{R}_a$ and $\mathcal{R}_D$. Then, in view of (H5) and (H8), $\mathbf{K}_a\varphi \in H(t)$; hence, by (H2), $\varphi \in H(t)$.

Assume, next, that $\neg\mathbf{K}_a\varphi \in H(s)$. In view of the inductive hypothesis, it suffices to show that there exist $t \in S$ such that $(s,t) \in \mathcal{R}'_a$ and $\neg\varphi \in H(t)$. By (H4), there exists $t \in S$ such that $(s,t) \in \mathcal{R}_a$ and $\neg\varphi \in H(t)$. As $\mathcal{R}_a \subseteq \mathcal{R}'_a$, the desired conclusion follows.

The case of $\chi = \mathbf{D}\varphi$ is very similar to the previous one and is left to the reader.

Suppose now that χ is $\mathbf{C}\varphi$. Assume that $\mathbf{C}\varphi \in H(s)$. In view of the inductive hypothesis, it suffices to show that if $(s,t) \in \mathcal{R}'_C$, then $\varphi \in H(t)$. So, assume that $(s,t) \in \mathcal{R}'_C$, i.e., either $s = t$ or, for some $n \geq 1$, there exists a sequence of states $s = s_0, s_1, \ldots, s_{n-1}, s_n = t$ such that, for every $0 \leq i < n$, either there exists $a \in \Sigma$ such that $(s_i, s_{i+1}) \in \mathcal{R}_a$ or $(s_i, s_{i+1}) \in \mathcal{R}_D$. In the former case, the desired conclusion follows from (H2); in the latter, it follows from (H2), (H3), and (H8).

Assume, on the other hand, that $\neg\mathbf{C}\varphi \in H(s)$. Then, the desired conclusion follows from (H9), the fact that $\mathcal{R}_C \subseteq \mathcal{R}'_C$, and inductive hypothesis.

To finish the proof of the lemma, we convert $\mathcal{M}'$ into a MAEM $\mathcal{M}''$ in a truth-preserving way. To that end, we use a variation of the construction known as tree-unwinding (see, for example, [4]; first applied in the context of epistemic logics with the operator of distributed knowledge in [3] and [9]). The only difference between our construction and the standard tree-unwinding is that, in the tree we produce, all edges labeled by D (representing the tree's relation $\mathcal{R}_D^T$) also get labeled (unlike in the standard tree-unwinding) by *all* agents in Σ, too; all other transitions are labeled by single agents, as in the standard tree-unwinding. To obtain $\mathcal{M}''$, we take $\mathcal{R}''_D$ to be the reflexive, symmetric, and transitive closure of $\mathcal{R}_D^T$ and $\mathcal{R}''_a$, for every $a \in \Sigma$, to be the reflexive, symmetric, and transitive closure of $\mathcal{R}_a^T$; finally, we take $\mathcal{R}''_C$ to be the reflexive closure of $\bigcup_{a\in\Sigma}\mathcal{R}''_a$. It is routine to check that $\mathcal{M}''$ is bisimilar to $\mathcal{M}'$ and, therefore, satisfies θ at its root. To complete the proof, all we have to show is that $\mathcal{M}''$ is a MAEM; i.e., the equality $\mathcal{R}''_D = \bigcap_{a\in\Sigma}\mathcal{R}''_a$ holds. The left-to-right direction is immediate from the construction. For the right-to-left direction assume that $(s,t) \in \mathcal{R}''_a$ holds for every $a \in \Sigma$; i.e, there is an undirected path between s and t along $\mathcal{R}_a^T$ for every $a \in \Sigma$. As we are in a tree and Σ contains at least two agents, this is only possible if there is an undirected path between s and t along $\mathcal{R}_D^T$ since we only connected nodes of the tree by multiple agent relations if these nodes were connected by $\mathcal{R}_D^T$. Therefore, $(s,t) \in \mathcal{R}''_D$, as desired. □

Theorem 3.6 *Let $\theta \in \mathcal{L}$. Then, θ is satisfiable in a MAEM iff there exists a MAEHS for θ.*

Proof. Immediate from lemma 3.4 and lemma 3.5. □

4. Tableau procedure for MAEL(CD)

Traditionally, tableaux work by decomposing the formula whose satisfiability is being tested into "semantically simpler" formulae. In the classical propositional case, "semantically simpler" implies "smaller", which by itself guarantees termination of the procedure. Another feature of the tableau method for the classical propositional logic is that this decomposition into simpler formulae results in a simple tree, representing an exhaustive search for a model—or, to be more precise, a Hintikka set (the classical analogue of Hintikka structures)—for the input formula. If at least one leaf of the tree produces a Hintikka set for the input formula, the search has succeeded and the formula is pronounced satisfiable.

These two defining features of the classical tableau method do not emerge unscathed when the method is applied to logics containing fixed point operators, such as $\mathbf{C}$ (or, for example, the $\mathcal{U}$ and $\neg\Box$ operators of the linear-time temporal logic **LTL**). Firstly, decomposing (in accordance with the clauses in the definition of a fully expanded set above) of formulae of the form $\mathbf{C}\varphi$ produces formulae of the form $\mathbf{K}_a(\varphi \wedge \mathbf{C}\varphi)$, which are "semantically simpler", but not smaller than the original formula. Hence, we cannot take termination for granted and need to take special

precautions to guarantee it—in our tableaux, we do so by deploying prestates, whose role is to ensure that the whole construction is finite. Secondly, in the classical case, the only reason why it might turn out to be impossible to produce a Hintikka set for the input formula is that every attempt to build such a set results in a collection of formulae containing an inconsistency. In the case of **MAEL**(CD), there are other such reasons; the most important of them has to do with eventualities: semantically, the truth of an eventuality $\neg\mathbf{C}\varphi$ at state s of a model requires that there is a path form s to a state t satisfying $\neg\varphi$. The analogue of this semantic condition in the tableau we refer to as *realization of eventualities*. Apart from consistency requirement on a "good" tableau, all eventualities in such a tableau should be realized. (A third, more technical reason why a tableau might fail to represent a MAEHS will be mentioned in due course.)

4.1. Overview of the tableau procedure

In essence, the tableau procedure for testing a formula $\theta \in \mathcal{L}$ for satisfiability is an attempt to construct a non-empty graph $\mathcal{T}^\theta$, called a *tableau*, representing all possible MAEHSs for θ (in the sense made precise later on). If the attempt is successful, θ is pronounced satisfiable; otherwise, it is declared unsatisfiable.

The tableau procedure consists of three major phases: *construction phase*, *prestate elimination phase*, and *state elimination phase*. Accordingly, we have three types of tableau rules: construction rules, a prestate elimination rule, and state elimination rules. The procedure itself essentially specifies in what order and under what circumstances these rules should be applied.

During the construction phase, the construction rules are used to produce a directed graph $\mathcal{P}^\theta$— called the *pretableau* for θ—whose set of nodes properly contains the set of nodes of the tableau $\mathcal{T}^\theta$ that we are building. Nodes of $\mathcal{P}^\theta$ are sets of formulae, some of which, called *states*, are meant to represent states of a Hintikka structure, while others, called *prestates*, fulfill a purely technical role of to keeping $\mathcal{P}^\theta$ finite. During the prestate elimination phase, we create a smaller graph $\mathcal{T}_0^\theta$ out of $\mathcal{P}^\theta$, called the *initial tableau for* θ, by eliminating all prestates of $\mathcal{P}^\theta$ (and tweaking with its edges) since prestates have already fulfilled their function: as we are not going to add any more nodes to the graph built so far, the possibility of producing an infinite structure is no longer a concern. Lastly, during the state elimination phase, we remove from $\mathcal{T}_0^\theta$ all states, if any, that cannot be satisfied in any MAEHS, for one of the following three reasons: either the state is inconsistent, or it contains an unrealized eventuality, or it does not have all successors needed for its satisfaction. The elimination procedure results in a (possibly empty) subgraph $\mathcal{T}^\theta$ of $\mathcal{T}_0^\theta$, called the *final tableau for* θ. Then, if we have some state Δ in $\mathcal{T}^\theta$ containing θ, we declare θ satisfiable; otherwise, we declare it unsatisfiable.

4.2. Construction phase

At this phase, we build the pretableau $\mathcal{P}^\theta$ — a directed graph whose nodes are sets of formulae, coming in two varieties: *states* and *prestates*. States are meant to represent states of a MAEHS which the tableau attempts to construct, while prestates are "embryo states", which will in the course of the construction be "unwound" into states. Technically, states are fully expanded (recall definition 3.1), while prestates do not have to be so.

Moreover, $\mathcal{P}^\theta$ will contain two types of edges. As we have already mentioned, our tableaux attempt to produce a MAEHS for the input formula; in this attempt, they set in motion an exhaustive search for such a MAEHS. One type of edge, depicted by unmarked double arrows $\Longrightarrow$, will represent this exhaustive search dimension of our tableaux. Exhaustive search looks for all possible alternatives, and in our tableaux the alternatives will arise when we unwind prestates into states; thus, when we draw an unmarked arrow from a prestate Γ to states Δ and Δ' (depicted as $\Gamma \Longrightarrow \Delta$ and $\Gamma \Longrightarrow \Delta'$, respectively), this intuitively means that, in any MAEHS, a state satisfying Γ has to satisfy at least one of Δ and Δ'.

Given a set $\Gamma \subseteq \mathcal{L}$, we say that Δ is a *minimal fully expanded extension of* Γ if Δ is fully expanded, $\Gamma \subseteq \Delta$, and no Δ' is such that $\Gamma \subseteq \Delta' \subset \Delta$ and Δ' is fully expanded.

Our first construction rule, **(SR)**, tells us how to create states from prestates. (Throughout the presentation of the rules, the reader can refer to the example given below to see how they are applied in particular cases.)

(SR) Given a prestate Γ, do the following:

1. add to the pretableau all minimal fully expanded extensions Δ of Γ as *states*;
2. for each so obtained state Δ, put $\Gamma \Longrightarrow \Delta$;
3. if, however, the pretableau already contains a state Δ' that coincides with Δ, do not create another copy of Δ', but only put $\Gamma \Longrightarrow \Delta'$.

We denote the finite set of states created by applying **(SR)** to a prestate Γ by $\mathbf{states}(\Gamma)$.

The second type of edge featuring in our tableaux represents accessibility relations in MAEHSs. Accordingly, this type of edge will be represented by single arrows marked with formulas whose presence in the source state requires the existence of a target state reachable by a particular relation. As there are two such kinds of formulae, $\neg\mathbf{K}_a\varphi$ and $\neg\mathbf{D}\varphi$ (see conditions (H4) and (H7) in the definition of MAEHS), we will have single arrows marked by formulas of one of these two types. Intuitively if, say $\neg\mathbf{K}_a\varphi \in \Delta$,

then we need some prestate Γ containing $\neg\varphi$ to be accessible by a relation $\mathcal{R}_a$; however, we mark this single arrow not just by agent a, but by formula $\neg\mathbf{K}_a\varphi$, which helps us remember not just what relation connects states satisfying Δ and Γ, but why we had to create this particular Γ. This information will prove crucial when we start eliminating prestates and then states.

The two remaining construction rules, **(KR)** and **(DR)**, tell us how to create prestates from states. These rules do not apply to patently inconsistent states as such states can not be satisfied in any MAEHS.

(KR) Given a state Δ such that $\neg\mathbf{K}_a\varphi \in \Delta$, for some $a \in \Sigma$, and there is no $\chi \in \mathcal{L}$ such that both $\chi \in \Delta$ and $\neg\chi \in \Delta$, do the following:

1. create a new prestate $\Gamma = \{\neg\varphi\} \cup \{\, \mathbf{K}_a\psi \mid \mathbf{K}_a\psi \in \Delta \,\} \cup \{\, \neg\mathbf{K}_a\psi \mid \neg\mathbf{K}_a\psi \in \Delta \,\}$;
2. connect Δ to Γ with $\stackrel{\neg\mathbf{K}_a\varphi}{\longrightarrow}$;
3. if, however, the tableau already contains a prestate $\Gamma' = \Gamma$, do not add to it another copy of Γ', but simply connect Δ to Γ' with $\stackrel{\neg\mathbf{K}_a\varphi}{\longrightarrow}$.

(DR) Given a state Δ such that $\neg\mathbf{D}\varphi \in \Delta$ and there is no $\chi \in \mathcal{L}$ such that both $\chi \in \Delta$ and $\neg\chi \in \Delta$, do the following:

1. create a new prestate $\Gamma = \{\neg\varphi\} \cup \{\, \mathbf{D}\psi \mid \mathbf{D}\psi \in \Delta \,\} \cup \{\, \neg\mathbf{D}\psi \mid \neg\mathbf{D}\psi \in \Delta \,\} \cup \{\, \mathbf{K}_a\chi \mid \mathbf{K}_a\chi \in \Delta, a \in \Sigma \,\} \cup \{\, \neg\mathbf{K}_a\chi \mid \neg\mathbf{K}_a\chi \in \Delta, a \in \Sigma \,\}$;
2. connect Δ to Γ with $\stackrel{\neg\mathbf{D}\varphi}{\longrightarrow}$;
3. if, however, the tableau already contains a prestate $\Gamma' = \Gamma$, do not add to it another copy of Γ', but simply connect Δ to Γ' with $\stackrel{\neg\mathbf{D}\varphi}{\longrightarrow}$.

It should be noted that, in the pretableau, we never create in one go full-fledged successors for states; i.e., we never draw a marked arrow from state to state; such arrows always go from states to prestates. On the other hand, unmarked arrows connect prestates to states.

When building a tableau for a formula θ, the construction stage starts off with creating a single prestate $\{\theta\}$. Afterwards, we alternate between applying rules creating states and those creating prestates: first, **(SR)** is applied to the prestates created at the previous stage of the construction, then **(KR)** and **(DR)** are applied to the states created at the previous stage. The construction phase comes to an end when every prestate required to be added to the pretableau has already been added (as prescribed in point 3 of **(SR)**), or when we end up with states to which neither **(KR)** nor **(DR)** is applicable (i.e. states not containing formulas of the form $\neg\mathbf{K}_a\varphi$ or $\neg\mathbf{D}\varphi$ or containing patent inconsistencies).

4.3. Termination of construction phase

As we identify states and prestates whenever possible, to prove that the above procedure terminates, it suffices to establish that there are only finitely many possible states and prestates. To that end we use the concept of the extended closure of a formula θ.

Definition 4.1 *Let $\theta \in \mathcal{L}$. The* closure *of θ, denoted $\mathsf{cl}(\theta)$, is the least set of formulae such that:*

- $\theta \in \mathsf{cl}(\theta)$*;*
- $\mathsf{cl}(\theta)$ *is closed under subformulae;*
- *if $\mathbf{K}_a\varphi \in \mathsf{cl}(\theta)$ for some $a \in \Sigma$, then $\mathbf{D}\varphi \in \mathsf{cl}(\theta)$;*
- *if $\mathbf{C}\varphi \in \mathsf{cl}(\theta)$, then $\mathbf{K}_a(\varphi \wedge \mathbf{C}\varphi) \in \mathsf{cl}(\theta)$ for every $a \in \Sigma$.*

Definition 4.2 *Let $\theta \in \mathcal{L}$. The* extended closure *of θ, denoted $\mathsf{ecl}(\theta)$, is the least set such that if $\varphi \in \mathsf{cl}(\theta)$, then $\varphi, \neg\varphi \in \mathsf{ecl}(\theta)$.*

It is straightforward to check that $\mathsf{ecl}(\theta)$ if finite for every θ and that all state and prestates of $\mathcal{P}^\theta$ are subsets of $\mathsf{ecl}(\theta)$; hence, their number is finite.

4.4. Prestate elimination phase

At this phase of the tableau procedure, we remove from $\mathcal{P}^\theta$ all prestates and all unmarked arrows, by applying the following rule:

(PR) For every prestate Γ in $\mathcal{P}^\theta$, do the following:

1. remove Γ from $\mathcal{P}^\theta$;
2. if there is a state Δ in $\mathcal{P}^\theta$ with $\Delta \stackrel{\chi}{\longrightarrow} \Gamma$, then for every state $\Delta' \in \mathbf{states}(\Gamma)$, put $\Delta \stackrel{\chi}{\longrightarrow} \Delta'$;

We call the graph obtained by applying **(PR)** to $\mathcal{P}^\theta$ the *initial tableau*, denoted by $\mathcal{T}_0^\theta$.

4.5. State elimination phase

During this phase, we remove from $\mathcal{T}_0^\theta$ nodes that cannot be satisfied in any MAEHS. There are three reasons why a state Δ of $\mathcal{T}_0^\theta$ can turn out to be unsatisfiable: Δ contains an inconsistency, *or* satisfiability of Δ requires satisfiability of some other unsatisfiable "successor" states, *or* Δ contains an eventuality that is not realized in the tableau. Accordingly, we have three elimination rules, **(E1)**–**(E3)**.

Technically, the state elimination phase is divided into stages; at stage $n+1$ we remove from the tableau $\mathcal{T}_n^\theta$ obtained at the previous stage exactly one state, by applying one of the elimination rules, thus obtaining the tableau $\mathcal{T}_{n+1}^\theta$. We now state the rules governing the process. The set of states of the tableau $\mathcal{T}_m^\theta$ is denoted by S_m^θ.

(E1) If $\{\varphi, \neg\varphi\} \subseteq \Delta \in S^\theta_n$, then obtain $\mathcal{T}^\theta_{n+1}$ by eliminating Δ from $\mathcal{T}^\theta_n$.

(E2) If Δ contains a formula χ of the form $\neg\mathbf{K}_a\varphi$ or $\neg\mathbf{D}\varphi$ and all states reachable from Δ by single arrows marked by χ have been eliminitated at previous stages, obtain $\mathcal{T}^\theta_{n+1}$ by eliminating Δ from $\mathcal{T}^\theta_n$.

To formulate the third elimination rule, we need the concept of eventuality realization. We say that $\neg\mathbf{C}\varphi$ is realized at Δ in $\mathcal{T}^\theta_n$ if there exists a path $\Delta = \Delta_0, \Delta_1, \ldots, \Delta_m$ such that $\neg\varphi \in \Delta_m$ and, for every $0 \leq i < m$, there exist χ such that $\Delta_i \xrightarrow{\chi} \Delta_{i+1}$.

Realization of eventuality $\neg\mathbf{C}\varphi$ at Δ in $\mathcal{T}^\theta_n$ can be easily checked by computing the *rank* of every $\Delta \in S^\theta_n$ with respect to $\neg\mathbf{C}\varphi$ in $\mathcal{T}^\theta_n$, denoted by $\mathbf{rank}(\Delta, \neg\mathbf{C}\varphi, \mathcal{T}^\theta_n)$. Intuitively, the rank of Δ in $\mathcal{T}^\theta_n$ represents the length of the longest path in $\mathcal{T}^\theta_n$ from Δ to a state containing $\neg\varphi$. If no such path exists, the rank of Δ is ω (the first infinite ordinal). Formally, the rank is computed as follows. At first, if $\neg\varphi \in \Delta$, set $\mathbf{rank}(\Delta, \neg\mathbf{C}\varphi, \mathcal{T}^\theta_n) = 0$; otherwise, set $\mathbf{rank}(\Delta, \neg\mathbf{C}\varphi, \mathcal{T}^\theta_n) = \omega$. Afterwards, repeat the following procedure until no changes in the rank of any state occurs: $\mathbf{rank}(\Delta, \neg\mathbf{C}\varphi, \mathcal{T}^\theta_n) = 1 + \max\{r_\chi\}$, where $r_\chi = \min\{\, \mathbf{rank}(\Delta', \neg\mathbf{C}\varphi, \mathcal{T}^\theta_n) \mid \Delta \xrightarrow{\chi} \Delta' \,\}$. Now, we can state our last rule.

(E3) If $\Delta \in S^\theta_n$ contains an eventuality $\neg\mathbf{C}\varphi$ that is not realized at Δ in $\mathcal{T}^\theta_n$ (i.e., if $\mathbf{rank}(\Delta, \neg\mathbf{C}\varphi, \mathcal{T}^\theta_n) = \omega$), then obtain $\mathcal{T}^\theta_{n+1}$ by removing Δ from $\mathcal{T}^\theta_n$.

We have thus far described the individual rules; to describe the state elimination phase as a whole, it is crucial to specify the order of their application.

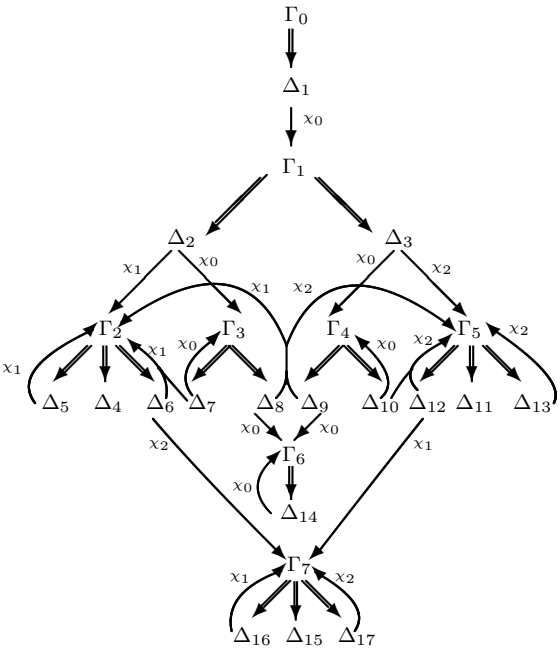
First, we apply **(E1)** to all states of $\mathcal{T}^\theta_0$; it is clear that, once this is done, we do not need to go back to **(E1)** again. The cases of **(E2)** and **(E3)** are slightly more involved. Having applied **(E3)** to the states of the tableau, we could have removed, for some Δ, all states accessible from it along the arrows marked with some formula χ; hence, we need to reapply **(E2)** to the resultant tableau to get rid of such Δ's. Conversely, having applied **(E2)**, we could have removed some states that were instrumental in realizing certain eventualities; hence, having applied **(E2)**, we need to reapply **(E3)**. Furthermore, we can't stop the procedure unless we have checked that *all* eventualities are realized. Thus, what we need is to apply **(E3)** and **(E2)** in a dovetailed sequence that cycles through all eventualities. More precisely, we arrange all eventualities occurring in the tableau obtained from $\mathcal{T}^\theta_0$ after having applied **(E1)** in the list $\xi_1, \ldots, \xi_m$. Then, we proceed in cycles. Each cycle consists of alternatingly applying **(E3)** to the pending eventuality, and then applying **(E2)** to the tableau resulting from that application, until all eventualities have been dealt with; once we reach ξ_m, we loop back to ξ_1. The cycles are repeated until, having gone through the whole cycle, we have not removed any states.

Once that happens, the state elimination phase is over. We call the resultant graph the *final tableau for* θ and denote it by $\mathcal{T}^\theta$ and its set of states by S^θ.

Definition 4.3 *The final tableau $\mathcal{T}^\theta$ is* open *if $\theta \in \Delta$ for some $\Delta \in S^\theta$; otherwise, $\mathcal{T}^\theta$ is* closed.

The tableau procedure returns "no" if the final tableau is closed; otherwise, it returns "yes" and, moreover, provides sufficient information for producing a finite model satisfying θ; that construction is described in section 5.2.

Example 1 *Let's assume that $\Sigma = \{a, b\}$ and construct a tableau for the formula $\mathbf{K}_a p \wedge \mathbf{K}_b p \wedge \neg\mathbf{D}\mathbf{C}p$. The picture below shows the complete pretableau for this formula.*



$\chi_0 = \neg\mathbf{D}\mathbf{C}p$, $\chi_1 = \neg\mathbf{K}_a(p \wedge \mathbf{C}p)$, $\chi_2 = \neg\mathbf{K}_b(p \wedge \mathbf{C}p)$;
$\Gamma_0 = \{\mathbf{K}_a p \wedge \mathbf{K}_b p \wedge \neg\mathbf{D}\mathbf{C}p\}$;
$\Delta_1 = \{\mathbf{K}_a p \wedge \mathbf{K}_b p \wedge \neg\mathbf{D}\mathbf{C}p, \mathbf{K}_a p, \mathbf{K}_b p, \neg\mathbf{D}\mathbf{C}p, \mathbf{D}p, p\}$;
$\Gamma_1 = \{\neg\mathbf{C}p, \mathbf{K}_a p, \mathbf{K}_b p, \neg\mathbf{D}\mathbf{C}p, \mathbf{D}p\}$;
$\Delta_2 = \{\neg\mathbf{C}p, \mathbf{K}_a p, \mathbf{K}_b p, \neg\mathbf{D}\mathbf{C}p, \mathbf{D}p, p, \neg\mathbf{K}_a(p \wedge \mathbf{C}p)\}$;
$\Delta_3 = \{\neg\mathbf{C}p, \mathbf{K}_a p, \mathbf{K}_b p, \neg\mathbf{D}\mathbf{C}p, \mathbf{D}p, p, \neg\mathbf{K}_b(p \wedge \mathbf{C}p)\}$;
$\Gamma_2 = \{\neg(p \wedge \mathbf{C}p), \mathbf{K}_a p, \neg\mathbf{K}_a(p \wedge \mathbf{C}p)\}$;
$\Gamma_3 = \{\neg\mathbf{C}p, \mathbf{K}_a p, \mathbf{K}_b p, \neg\mathbf{D}\mathbf{C}p, \mathbf{D}p, \neg\mathbf{K}_a(p \wedge \mathbf{C}p)\}$;
$\Delta_4 = \{\neg p, \mathbf{K}_a p, \neg\mathbf{K}_a(p \wedge \mathbf{C}p), \mathbf{D}p, p\}$;
$\Delta_5 = \{\neg\mathbf{C}p, \mathbf{K}_a p, \neg\mathbf{K}_a(p \wedge \mathbf{C}p), \mathbf{D}p, p\}$;
$\Delta_6 = \{\neg\mathbf{C}p, \mathbf{K}_a p, \neg\mathbf{K}_a(p \wedge \mathbf{C}p), \mathbf{D}p, p, \neg\mathbf{K}_b(p \wedge \mathbf{C}p)\}$;
$\Delta_7 = \{\neg\mathbf{C}p, \mathbf{K}_a p, \mathbf{K}_b p, \neg\mathbf{D}\mathbf{C}p, \mathbf{D}p, \neg\mathbf{K}_a(p \wedge \mathbf{C}p), p\}$;
$\Delta_8 = \{\neg\mathbf{C}p, \mathbf{K}_a p, \mathbf{K}_b p, \neg\mathbf{D}\mathbf{C}p, \mathbf{D}p, \neg\mathbf{K}_a(p \wedge \mathbf{C}p), p, \neg\mathbf{K}_b(p \wedge \mathbf{C}p)\}$
$\Gamma_4 = \{\neg\mathbf{C}p, \mathbf{K}_a p, \mathbf{K}_b p, \neg\mathbf{D}\mathbf{C}p, \mathbf{D}p, \neg\mathbf{K}_b(p \wedge \mathbf{C}p)\}$
$\Gamma_5 = \{\neg(p \wedge \mathbf{C}p), \mathbf{K}_b p, \neg\mathbf{K}_b(p \wedge \mathbf{C}p)\}$
$\Delta_9 = \{\neg\mathbf{C}p, \mathbf{K}_a p, \mathbf{K}_b p, \neg\mathbf{D}\mathbf{C}p, \mathbf{D}p, \neg\mathbf{K}_b(p \wedge \mathbf{C}p), \neg\mathbf{K}_a(p \wedge \mathbf{C}p)\}$
$\Delta_{10} = \{\neg\mathbf{C}p, \mathbf{K}_a p, \mathbf{K}_b p, \neg\mathbf{D}\mathbf{C}p, \mathbf{D}p, \neg\mathbf{K}_b(p \wedge \mathbf{C}p), p\}$

$\Delta_{11} = \{\neg p, \mathbf{K}_b p, \neg \mathbf{K}_b(p \wedge \mathbf{C}p), \mathbf{D}p, p\}$
$\Delta_{12} = \{\neg \mathbf{C}p, \mathbf{K}_b p, \neg \mathbf{K}_b(p \wedge \mathbf{C}p), \neg \mathbf{K}_a(p \wedge \mathbf{C}p), \mathbf{D}p, p\}$
$\Delta_{13} = \{\neg \mathbf{C}p, \mathbf{K}_b p, \neg \mathbf{K}_b(p \wedge \mathbf{C}p), \mathbf{D}p, p\}$
$\Gamma_6 = \{\neg \mathbf{C}p, \mathbf{K}_a p, \mathbf{K}_b p, \neg \mathbf{D}\mathbf{C}p, \mathbf{D}p, \neg \mathbf{K}_a(p \wedge \mathbf{C}p), \neg \mathbf{K}_b(p \wedge \mathbf{C}p)\}$
$\Delta_{14} = \{\neg \mathbf{C}p, \mathbf{K}_a p, \mathbf{K}_b p, \neg \mathbf{D}\mathbf{C}p, \mathbf{D}p, \neg \mathbf{K}_a(p \wedge \mathbf{C}p), \neg \mathbf{K}_b(p \wedge \mathbf{C}p), p\}$
$\Gamma_7 = \{\neg(p \wedge \mathbf{C}p)\}$
$\Delta_{15} = \{\neg p\}$;
$\Delta_{16} = \{\neg \mathbf{C}p, \neg \mathbf{K}_a(p \wedge \mathbf{C}p\}$;
$\Delta_{17} = \{\neg \mathbf{C}p, \neg \mathbf{K}_b(p \wedge \mathbf{C}p\}$

*For lack of space, we do not depict the initial and final tableaux for the input formula, but briefly describe what happens at the state elimination stage. States Δ_4 and Δ_{11} get removed due to **(E1)**, as they contain patent inconsistencies. Δ_{14} gets removed due to **(E3)**, since it contains an eventuality $\neg \mathbf{C}p$ which is not realized in the tableau, as the rank of Δ_{14} stabilizes at ω, because it does not contain $\neg p$, and is its only successor. Then Δ_8 and Δ_9 get removed, as their only successor along χ_0, namely Δ_{14} has been removed. All other states remain in place; in particular, all of them receive a finite rank, because from each of them one can reach the state Δ_{15}, which contains $\neg p$. The resultant graph encodes all possible Hintikka structures for the input formula.*

We note that our tableaux never close on account of all states obtained from the initial prestate containing unfulfilled eventualities (we omit the formal proof of this claim due to lack of space). The rule **(E3)**, however, as can be seen from the example above, eliminates from the tableau "bad" states, thus making our tableau not only test a formula for satisfiability, but actually, for every satisfiable formula θ, produce a graph "containing" all possible Hintikka structures for θ (i.e, whenever a node of the graph is connected to several other nodes by arrows marked by the same formula, these "target" nodes are not meant to be part of the same MAEHS for θ, but rather represent alternative ways of building a MAEHS for θ).

5. Soundness and completeness

5.1. Soundness

The soundness of a tableau procedure amounts to claiming that if the input formula θ is satisfiable, then the tableau for θ is open. To establish soundness of the overall procedure, we prove a series of lemmas that show that every rule is sound; the soundness of the overall procedure will then easily follow. The proofs of the following three lemmas are straightforward.

Lemma 5.1 *Let Γ be a prestate of $\mathcal{P}^\theta$ such that $\mathcal{M}, s \Vdash \Gamma$ for some MAEM $\mathcal{M}$ and $s \in \mathcal{M}$. Then, $\mathcal{M}, s \Vdash \Delta$ holds for at least one $\Delta \in \mathbf{states}(\Gamma)$.*

Lemma 5.2 *Let $\Delta \in S_0^\theta$ be such that $\mathcal{M}, s \Vdash \Delta$ for some MAEM $\mathcal{M}$ and $s \in \mathcal{M}$, and let $\neg \mathbf{K}_a \varphi \in \Delta$. Then, there exists $t \in \mathcal{M}$ such that $(s,t) \in \mathcal{R}_a$ and $\mathcal{M}, t \Vdash \{\neg\varphi\} \cup \{\, \mathbf{K}_a\psi \mid \mathbf{K}_a\psi \in \Delta \,\} \cup \{\, \neg\mathbf{K}_a\psi \mid \neg\mathbf{K}_a\psi \in \Delta \,\}$.*

Lemma 5.3 *Let $\Delta \in S_0^\theta$ be such that $\mathcal{M}, s \Vdash \Delta$ for some MAEM $\mathcal{M}$ and $s \in \mathcal{M}$, and let $\neg \mathbf{D}\varphi \in \Delta$. Then, there exists $t \in \mathcal{M}$ such that $(s,t) \in \mathcal{R}_D$ and $\mathcal{M}, t \Vdash \{\neg\varphi\} \cup \{\, \mathbf{D}\psi \mid \mathbf{D}\psi \in \Delta \,\} \cup \{\, \neg\mathbf{D}\psi \mid \neg\mathbf{D}\psi \in \Delta \,\} \cup \{\, \mathbf{K}_a\chi \mid \mathbf{K}_a\chi \in \Delta, a \in \Sigma \,\} \cup \{\, \neg\mathbf{K}_a\chi \mid \neg\mathbf{K}_a\chi \in \Delta, a \in \Sigma \,\}$.*

Lemma 5.4 *Let $\Delta \in S_0^\theta$ be such that $\mathcal{M}, s \Vdash \Delta$ for some MAEM $\mathcal{M}$ and $s \in \mathcal{M}$, and let $\neg \mathbf{C}\varphi \in \Delta$. Then, $\neg \mathbf{C}\varphi$ is realized at Δ in $\mathcal{T}_n^\theta$.*

Proof. As Δ is fully expanded, $\neg \mathbf{K}_a(\varphi \wedge \mathbf{C}\varphi) \in \Delta$ for some $a \in \Sigma$, and thus $\mathcal{M}, s \Vdash \neg \mathbf{K}_a(\varphi \wedge \mathbf{C}\varphi)$. Therefore, there exists $s_1 \in \mathcal{M}$ such that $(s, s_1) \in \mathcal{R}_a$ and $\mathcal{M}, s_1 \Vdash \neg(\varphi \wedge \mathbf{C}\varphi)$. By construction of the tableau, $\mathcal{M}, s_1 \Vdash \Gamma$ holds for the prestate Γ associated with $\neg \mathbf{K}_a(\varphi \wedge \mathbf{C}\varphi)$, i.e. such Γ that $\neg(\varphi \wedge \mathbf{C}\varphi) \in \Gamma$. Now, there exists $\Delta_1 \in \mathbf{states}(\Gamma)$ such that $\mathcal{M}, s_1 \Vdash \Delta_1$. Indeed, elements of $\mathbf{states}(\Gamma)$ are full expansions of Γ; clearly, Γ can be fully expanded in such a way that whenever we have to make a choice which of several formulae to include into Δ_1 (say, for which $b \in \Sigma$ to add the formula $\neg \mathbf{K}_b(\varphi \wedge \mathbf{C}\varphi)$ if $\neg \mathbf{C}\varphi \in \Gamma$), we choose the one that is actually satisfied at s_1. Now, as $\neg(\varphi \wedge \mathbf{C}\varphi) \in \Gamma$, either $\mathcal{M}, s_1 \Vdash \neg\varphi$ or $\mathcal{M}, s_1 \Vdash \neg \mathbf{C}\varphi$. In the former case, we are done straight off, as then $\neg\varphi \in \Delta_1$. In the latter case, as $\mathcal{M}, s_1 \Vdash \neg \mathbf{C}\varphi$, there exists a sequence of states $s_1, s_2, \ldots, s_m$ in $\mathcal{M}$ such that for every $1 \leq i < m$, we have $(s_i, s_{i+1}) \in \mathcal{R}_b$ for some $b \in \Sigma$ and $\mathcal{M}, s_m \Vdash \neg\varphi$. By taking this sequence of states of $\mathcal{M}$, we can build, in the "forcing choices" style described above, a sequence of states $\Delta_1, \Delta_2, \ldots, \Delta_m \in S_n^\theta$ such that, for every $1 \leq i < m$, we have $\Delta_i \stackrel{\neg \mathbf{K}_b(\varphi \wedge \mathbf{C}\varphi)}{\longrightarrow} \Delta_{i+1}$ for some $b \in \Sigma$, and $\neg\varphi \in \Delta_m$. The existence of the path $\Delta, \Delta_1, \ldots, \Delta_m$ implies that $\neg \mathbf{C}\varphi$ is realized at Δ in $\mathcal{T}_n^\theta$. □

Theorem 5.5 (Soundness) *If $\theta \in \mathcal{L}$ is satisfiable in a MAEM, then $\mathcal{T}^\theta$ is open.*

Proof sketch. Using the preceding lemmas, show by induction on the number of stages in the state elimination process that no satisfiable state can be eliminated due to **(E1)**–**(E3)**. The claim then follows from lemma 5.1. □

5.2. Completeness

The completeness of a tableau procedure means that if the tableau for a formula θ is open, then θ is satisfiable in a MAEM. By making use of theorem 3.6, it suffices to show

that an open tableau for θ can be turned into a MAEHS for θ. The construction of such a MAEHS is described in the following lemma.

Lemma 5.6 *If $\mathcal{T}^\theta$ is open, then there exists a MAEHS for θ.*

Proof sketch. Let $\mathcal{T}^\theta$ be open. The MAEHS $\mathcal{H}$ for θ is built out of the so-called *final tree components*. Each final tree component is a tree-like MAES with nodes labeled with states from S^θ. Each component is associated with a state $\Delta \in S^\theta$ and an eventuality $\xi \in \mathsf{ecl}(\theta)$; such a component is denoted by $T_{\Delta,\xi}$.

Now we describe how to build the final tree components. Let $\xi = \neg\mathbf{C}\varphi \in \mathsf{ecl}(\theta)$ and $\Delta \in S^\theta$. If $\xi \notin \Delta$, then $T_{\Delta,\xi}$ is a "simple tree" (i.e, one whose only inner node is the root) whose root is labeled with Δ and that has exactly one leaf associated with each formula of the form $\neg\mathbf{K}_a\varphi$ or $\neg\mathbf{D}\varphi$ belonging to Δ. A leaf associated with formula χ is labeled by a state $\Delta' \in S^\theta$ such that in $\mathcal{T}^\theta$ we have $\Delta \xrightarrow{\chi} \Delta'$ (such a Δ' exists—otherwise Δ would have been eliminated from the tableau due to **(E2)**). To obtain a tree-like MAES, put $(s,t) \in \mathcal{R}_a$ if s is labeled with Δ, t is labeled with Δ', and $\Delta \xrightarrow{\neg\mathbf{K}_a\varphi} \Delta'$ for some φ; analogously, put $(s,t) \in \mathcal{R}_D$ if s is labeled with Δ, t is labeled with Δ', and $\Delta \xrightarrow{\neg\mathbf{D}\varphi} \Delta'$ for some φ.

If, on the other hand, $\xi = \neg\mathbf{C}\varphi \in \Delta$, then $T_{\Delta,\xi}$ is constructed as follows. Since $\neg\mathbf{C}\varphi$ is realized at Δ in $\mathcal{T}^\theta$, there exists a sequence of states $\Delta = \Delta_0, \Delta_1, \ldots, \Delta_m$ in S^θ such that $\neg\varphi \in \Delta_m$ and for every $0 \leq i < m$, $\Delta \xrightarrow{\chi} \Delta'$ holds for some χ of the form $\neg\mathbf{K}_a\varphi$ or $\neg\mathbf{D}\varphi$ (otherwise, it would have been eliminated due to **(E3)**). Take this sequence and give to each Δ_i ($0 \leq i \leq m$) "enough" successors, as in the previous paragraph, and define the relations for this tree as prescribed therein.

We are next going to stitch the above-defined $T_{\Delta,\xi}$'s together. First, however, we note that if an eventuality ξ' belongs to Δ and is not realized inside some final tree component $T_{\Delta,\xi}$ (the realization in a final tree component is defined as in tableaux, with substituting $T_{\Delta,\xi}$ for $\mathcal{T}^\theta_n$), then ξ' belongs to every leaf of $T_{\Delta,\xi}$, and thus its realization is deferred—this is crucial to our ability to stitch $T_{\Delta,\xi}$'s up into a Hintikka structure.

We now proceed as follows. First, we arrange all states of $\mathcal{T}^\theta$ in a list $\Delta_0, \ldots, \Delta_{n-1}$ and all eventualities occurring in the states of $\mathcal{T}^\theta$ in a list $\xi_0, \ldots, \xi_{m-1}$. We then think of all final tree components as arranged in an m-by-n grid whose rows are marked with the correspondingly numbered eventualities of $\mathcal{T}^\theta$ and whose columns are marked with the correspondingly numbered states of $\mathcal{T}^\theta$. The final tree component at the intersection of the ith row and the jth column will be denoted by $T_{(i,j)}$. The building blocks for our MAEHS will all come from the grid. This MAEHS is built incrementally, so that at each stage of the construction we produce a structure realizing more and more eventualities.

We start off with a final tree component that is uniquely determined by the input formula θ, in the following way. If θ is an eventuality, i.e., $\theta = \xi_p$ for some $0 \leq p < m$, then we start off with the component $T_{(p,q)}$ where, for definiteness, q is the least number $< n$ such that $\theta \in \Delta_q$; as $\mathcal{T}^\theta$ is open, such a q exists. If, on the other hand, θ is not an eventuality, then we start off with $T_{(0,q)}$, where q is as described above. Let's denote this initial structure by $\mathcal{H}_0$.

Henceforth, we proceed as follows. Informally, we think of the above list of eventualities as a queue of customers waiting to be served. Unlike the usual queues, we do not necessarily start serving the queue from the first customer (if θ is an eventuality, then it gets served first; otherwise we start from the beginning of the queue), but then we follow the queue order, curving back to the beginning of the queue after having served its last eventuality, if we started in the middle. Serving an eventuality ξ amounts to appending to the leaves of the structure built thus far final tree components realizing ξ. Thus, we keep track of what eventualities have already been served, take note of the one that was served the last, say ξ_j, and replace every leaf of the structure $\mathcal{H}_i$ constructed thus far with the final tree component $T_{i+1,((j+1) \bmod m)}$. The process continues until all eventualities have been served, at which point we have gone the full cycle through the queue.

After that, the cycle is repeated, for as long as the queue remains non-empty. Alternatively, if we want to guarantee that the MAEHS we are building is going to be finite, the cycle is repeated with the following modification: whenever the component we are about to attach, say $T_{(i,j)}$, is already contained in our structure in the making, instead of replacing the leaf t with that component, we connect every "predecessor" s of t to the root of $T_{(i,j)}$ with the relation connecting s to t. This modified version of the cycle is repeated until we come to a point when no more components get added—this is bound to happen in a finite number of steps as the number of $T_{\Delta,\xi}$'s is finite. It is now routine to check that the resultant structure $\mathcal{H}$ is a Hintikka structure, whose set of agents is the set of agents occurring in θ. By construction, it contains a node labeled with a set containing θ. □

Theorem 5.7 (Completeness) *Let $\theta \in \mathcal{L}$ and let $\mathcal{T}^\theta$ be open. Then, θ is satisfiable in a MAEM.*

Proof. Immediate from lemma 5.6 and theorem 3.6. □

6. Complexity of the procedure

Let's denote the length of the input formula θ by n and the number of agents in the language by k. We assume that $k > 1$, otherwise we just deal with the modal logic $\mathbf{S5}$. The size of the extended closure for θ (recall definition 4.2) is bounded from above by $\mathcal{O}(k^n)$, as each $\mathbf{C}$ operator occurring in θ requires k formulas to be added to the extended closure.

The examination of the procedure shows that the longest path to any state of the pretableau we create at the construction phase from the initial prestate (i.e., the one containing the input formula θ) is bound by the number of nested "diamond" modalities (such as $\neg\mathbf{K}_a$) in θ plus 1. From any given state or prestate we can create at most $\mathcal{O}(k^n)$ (pre-)states, hence the whole number of nodes we create is in $\mathcal{O}(k^{n^2})$. Thus, the construction phase can be done in time $\mathcal{O}(k^{n^2})$.

At the prestate elimination phase, we delete at most $\mathcal{O}(k^{n^2})$ states and for each prestate redirect at most $\mathcal{O}(k^n)$ arrows, which takes within $\mathcal{O}(k^{n^2})$ steps.

At the state elimination stage, we first apply **(E1)** to $\mathcal{O}(k^{n^2})$ states, which can be done in $\mathcal{O}(k^{(2n+n^2)})$ steps. After that, we embark on the dovetailed application of **(E2)** and **(E3)**. We proceed in circles, whose number is bound by $\mathcal{O}(k^{n^2})$, as at each iteration we remove at least one state. During each cycle, we carry out $\mathcal{O}(k^n)$ times (the upper bound on the number of eventualities) the following procedure: fist, we apply **(E2)** to all states, which can be done in time $\mathcal{O}(k^{(n+n^2)})$, and then apply **(E3)** to the pending eventuality. The latter procedure is carried out by computing a rank of each state of the tableau with respect to the pending eventuality. The number of rank updates is bound by $\mathcal{O}(k^{n^2})$, each update requiring $\mathcal{O}(k^{(n+n^2)})$ steps, as for each state Δ we check the ranks of the targets of outgoing arrows marked by formulae in Δ. Thus, the whole state elimination phase can be carried out in $\mathcal{O}(k^{2n^2})$ steps.

We conclude that the whole procedure can be carried out in $\mathcal{O}(k^{2n^2})$ steps, where n is the size of the input formula. It follows that **MAEL**(CD)-satisfiability is in $\mathbf{ExpTime}$, which together with the result from [7] implies that **MAEL**(CD)-satisfiability is $\mathbf{ExpTime}$-complete.

7 Concluding remarks

We have developed a sound, complete, and complexity-optimal incremental-tableau-based decision procedure for the multi-agent epistemic logic **MAEL**(CD). We claim that this style of tableau is of immediate practical use, both by human and computerized execution. It is more efficient (within the theoretically established complexity bounds) and more modular and adaptable than the top-down tableaux of the type developed (for a fragment of the logic not including the $\mathbf{D}$ operator) in [7]. In particular, the tableaux presented lends itself to an extension to the full multi-agent epistemic logic, with modal operators of common and distributed knowledge for all coalitions of agents, and well as to a combination with the similar style tableaux developed for the Alternating-time temporal logic **ATL** developed in [5], which are going to be the subject of our subsequent work.

Acknowledgments This research was supported by a research grant of the National Research Foundation of South Africa and was done during the second author's postdoctoral fellowship at the University of the Witwatersrand, funded by the Claude Harris Leon Foundation—we gratefully acknowledge the financial support from these institutions. We also acknowledge the anonymous referees whose remarks helped to improve our presentation.

References

[1] E. A. Emerson and J. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *Journal of Computation and System Sciences*, 30(1):1–24, 1985.

[2] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning about Knowledge*. MIT Press: Cambridge, MA, 1995.

[3] R. Fagin, J. Y. Halpern, and M. Y. Vardi. What can machines know? On the properties of knowledge in distributed systems. *Journal of the ACM*, 39(2):328–376, April 1992.

[4] V. Goranko and M. Otto. Model theory of modal logic. In *Handbook of Modal Logic*, pages 249–330. Elsevier, 2007.

[5] V. Goranko and D. Shkatov. Tableau-based decision procedures for logics of strategic ability in multi-agent systems. To appear in *ACM Transactions on Computational Logic*. Available at `http://arxiv.org/abs/0803.2306`.

[6] J. Y. Halpern and Y. Moses. Knowledge and common knowledge in a distributed environment. *Journal of ACM*, 37(3):549–587, 1990.

[7] 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.

[8] J.-J. C. Meyer and W. van der Hoek. *Epistemic Logic for Computer Science and Artificial Intelligence*. CUP, 1995.

[9] W. van der Hoek and J.-J. C. Meyer. Making some issues of implicit knowledge explicit. *International Journal of Foundations of Computer Science*, 3(2):193–224, 1992.

[10] W. van der Hoek and J.-J. C. Meyer. A complete epistemic logic for multiple agents–combining distributed and common knowledge. In M. O. L. B. et al., editor, *Epistemic Logic and the Theory of Games and Decisions*, pages 35–68. Kluwer Academic Publishers, 1997.

[11] P. Wolper. The tableau method for temporal logic: an overview. *Logique et Analyse*, 28(110–111):119–136, 1985.

[12] M. Wooldridge. *An Introduction to Multiagent Systems*. John Willey and Sons, 2002.