Tableau-based decision procedure for the multiagent epistemic logic with all coalitional operators for common and distributed knowledge

MAI AJSPUR*, Department of Communication, Business and Information Technologies, Roskilde University, Roskilde 4000, Denmark

VALENTIN GORANKO†, Department of Informatics and Mathematics, Technical University of Denmark, Kongens Lyngby 2800, Denmark, and Department of Mathematics, University of Johannesburg, Johannesburg 2006, South Africa

DMITRY SHKATOV‡, School of Computer Science, University of the Witwatersrand, Johannesburg 2050, South Africa

Abstract

We develop a conceptually clear, intuitive and feasible decision procedure for testing satisfiability in the full multiagent epistemic logic CMAEL(CD) with operators for common and distributed knowledge for all coalitions of agents mentioned in the language. To that end, we introduce Hintikka structures for CMAEL(CD) and prove that satisfiability in such structures is equivalent to satisfiability in standard models. Using that result, we design an incremental tableau-building procedure that eventually constructs a satisfying Hintikka structure for every satisfiable input set of formulæ of CMAEL(CD) and closes for every unsatisfiable input set of formulæ.

Keywords: multi-agent epistemic logic, satisfiability, tableau, decision procedure

1 Introduction

Over the last three decades, multiagent epistemic logics [9], [27] have been playing an increasingly important role in computer science and AI. The earliest prominent applications have been to specification, design and verification of distributed protocols [23] and [24]; a number of other applications are described in, among others, [9], [10], and [27]. The most recent, and perhaps more important ones are to specification, design and verification of multiagent systems—a research area that has emerged on the borderline between distributed computing, AI, and game theory [36], [45], [47].

1.1 Multiagent epistemic logics and decision methods for them

Languages of multiagent epistemic logics considered in the literature contain various repertoires of epistemic operators. We refer to the basic multiagent epistemic logic, containing only operators of

*E-mail: ajspur@ruc.dk
†E-mail: vgro@imm.dtu.dk
‡E-mail: dmitry@cs.wits.ac.za

© The Author 2012. Published by Oxford University Press. All rights reserved.
For Permissions, please email: journals.permissions@oup.com
individual knowledge for a finite non-empty set $\Sigma$ of agents, as $\text{MAEL}$ (Multi Agent Epistemic Logic). Since all epistemic operators of this logic are $\text{S5}$-type modalities, it is also referred to in the literature as $\text{S5}_n$, where $n$ is the number of agents in the language. The logic obtained from $\text{MAEL}$ by adding the operator of common knowledge among all agents in $\Sigma$ is then called $\text{MAEL(C)}$. This logic, along with $\text{MAEL}$, was studied in [25]. Analogously, if $\text{MAEL}$ is augmented with the operator of distributed knowledge for all agents, then the resulting logic will be called $\text{MAEL(D)}$. It was studied in [10] and [38]. $\text{MAEL}$ augmented with operators of both common and distributed knowledge for the set of all agents, hereafter called $\text{MAEL(CD)}$, was studied in [39], and a tableau-based decision procedure for it was first presented in [14]. Thus, all logics mentioned so far either do not have both operators of common and distributed knowledge, or only have those operators for the whole set of agents in the language.

At the same time, there has recently been an increasing interest in the study of coalitional multiagent logics (see [30], [31], [32], [2], [40], [13]), i.e. logics whose languages refer to any groups (coalitions) of agents. These are important, inter alia, in multiagent systems, where agents may ‘cooperate’ (i.e. form a coalition) in order to achieve a certain goal. Most of the so far studied logical formalisms referring to coalitions of agents have only been concerned with formalizing reasoning about strategic abilities of coalitions. (A notable exception is [44], where the Alternating-time Temporal Epistemic Logic $\text{ATEL}$ was introduced, whose language contains both common knowledge and strategic operators for coalitions of agents.) Clearly, real cooperation can only be achieved by communication, i.e. exchange of knowledge. Thus, it is particularly natural and important to consider multiagent epistemic logics with operators for both common and distributed knowledge among any (non-empty) coalitions of agents. This is the logic under consideration in the present article, hereby called $\text{CMAEL(CD)}$ (for Coalitional Multi-Agent Epistemic Logic with operators of Common and Distributed knowledge). It subsumes all multiagent epistemic logics mentioned above, except $\text{ATEL}$.

In order to be practically useful for such tasks as specification and design of distributed or multiagent systems, the respective logic need to be equipped with algorithms solving (constructively) its satisfiability problem, i.e. testing whether a given input formula $\varphi$ of that logic is satisfiable and, if so, providing enough information for the construction of a model for $\varphi$. Decidability of modal logics, including epistemic logics, is usually proved by establishing a ‘small model property’, which provides a brute force decision procedure consisting of exhaustive search for a model amongst all those whose size is within the theoretically prescribed bounds. The two most common practically feasible general methods for satisfiability checking of modal logics are based on automata [41] and on tableaux (see e.g. [3], [1], [4], [8], [19], [12]).

There are various styles of tableau-based decision procedures; see [1], [15] and [12] for detailed exposition and surveys. An easy to describe but somewhat less efficient and practically unfeasible approach, that we will call maximal tableau (also called top-down in [4]), consists in trying to build in one step a ‘canonical’ finite model for any given formula out of all maximal consistent subsets of the closure of that formula. This method always works in (at least) exponential time and usually produces a wastefully large model, if any exists. A more flexible and more practically applicable version, adopted in the present paper, is a so called incremental (aka, ‘bottom-up’) tableau building procedure. While in all known cases, the worst-case time complexity for maximal and incremental tableaux are the same, the crucial difference is that maximal tableaux always require the amount of resources predicted by the theoretical worst-case time estimate, while incremental tableaux work on average much more efficiently.\footnote{This claim cannot be made mathematically precise due to the lack of an a priori probability distribution on formulae of a logic. The interested reader may consult [13] for comparison of efficiency of the two types of tableaux in the context of Alternating-time temporal logic $\text{ATL}$.}
1.2 Related work and comparison

The present work is part of a series of papers ([17], [14], [18], [15]) where we have embarked on the project of developing practically efficient yet intuitive and conceptually clear incremental-tableau-based satisfiability checking procedures for a range of multiagent logics. This article builds on the conference papers [14] and [18] by substantially extending, revising and improving them.

There are three inherent complications affecting the construction of a tableau procedure for the logic \textbf{CMAEL(CD)}, arising respectively from the common knowledge (fixpoint-definable operator), the distributed knowledge (with associated epistemic relation being the intersection of the individual knowledge epistemic relations), and the interactions between the knowledge operators over different coalitions of agents.

Several tableau-based methods for satisfiability-checking for modal logics with fixpoint-definable operators have been developed and published over the past 30 years, all going back to the tableau-based decision methods developed for the Propositional Dynamic Logic \textbf{PDL} in [34], for the branching-time temporal logics \textbf{UB} in [3] and \textbf{CTL} in [8, Section 5] and [7]. In terms of handling eventualities arising from the fixed-point operators our tableau method follows more closely on the incremental tableaux for the linear time temporal logic \textbf{LTL} in [46] and for \textbf{CTL} in [8, Section 7].

A particular complication arising in the tableau for \textbf{CMAEL(CD)} stems from the fact that the epistemic operators, being \textbf{S5} modalities, are symmetric, and thus the epistemic boxes have global effect on the model, too. This requires a special mechanism for propagating their effect backwards when occurring in states of the tableau. In the present article, we have chosen to implement such mechanism by using \textit{analytic cut rules}, going back to Smullyan [37] and Fitting [11], see also [19] and [28]. More recently, tableaux with analytic cut rules for modal logics with symmetric relations have been developed in [21], [20], [5].

We note that there is a natural tradeoff between conceptual clarity and simplicity of (tableau-based) decision procedures on the one hand, and their technical sophistication and optimality on the other hand. We emphasize that the main objective of developing the tableau procedure presented here is the conceptual clarity, intuitiveness and ease of implementation, rather than practical optimality. While being optimal in terms of worst-case time complexity and incorporating some new and non-trivial optimizing features (such as restricted applications of cut rules) this procedure is amenable to various improvements and further optimizations. Most important known such optimizations are \textit{on-the-fly} techniques for elimination of bad states and \textit{one-pass} tableau methods developed for some related logics in [33], [1] and \textit{cut-free} versions of tableau as in [1] for \textbf{MAEL(C)}, [2] for \textbf{PDL} with converse operators, [29] for the description logic \textbf{SHI} and of sequent calculi, in [26] for \textbf{MAEL(C)} and in [24] for \textbf{LTL} and \textbf{CTL}. We discuss briefly the possible modifications of our procedure, implementing such optimizing techniques in Section 6.

Here is a summary (in a roughly chronological order) of the more closely related previous work, besides our own, on tableau-based decision procedures for multiagent epistemic logics with common and/or distributed knowledge:

- the maximal tableaux for \textbf{MAEL(C)} presented in [23];
- the semantic construction used in [14, Appendix A1] to prove completeness of an axiomatic system for \textbf{MAEL(D)};
- the proof of decidability of \textbf{MAEL(CD)} based on finite model property via filtration in [33];
- the maximal tableau-like decision procedure for \textbf{ATL}, presented in [44] and extended to \textbf{ATEL} in [43].
Tableaux for the full coalitional multiagent epistemic logic

- the exponential-time tableau-based procedure developed in [6] for testing satisfiability in the BDI logic, that has some common features with CMAEL(C);
- the optimized cut-free single-pass tableaux for the multi-agent logic of common knowledge MAEL(C), in [1]. on tableaux for multiagent logics using global caching and analytic cuts in [3].

1.3 Structure of the paper

In Section 2, we introduce the syntax and semantics of the logic CMAEL(CD). In Section 3, we introduce Hintikka structures for CMAEL(CD) and show that Hintikka structures are equivalent to Kripke models with respect to satisfiability of formulae. Then, in Section 4, we develop the tableau procedures checking for satisfiability of formulae of CMAEL(CD). In Section 5, we prove the correctness of our procedure in Section 6 we estimate its complexity, discuss it efficiency and indicate some possible technical improvements. We end with concluding remarks pointing out some directions for further development.

2 Syntax and semantics

2.1 Syntax of CMAEL(CD)

The language of CMAEL(CD) contains a fixed, at most countable, set AP of atomic propositions, typically denoted by p, q, r, ...; a finite, non-empty set Σ of (names for) agents typically denoted by a, b, ..., while sets of agents, called coalitions, will be usually denoted by A, B, ...; a sufficient repertoire of the Boolean connectives, say ¬ ('not') and ∧ ('and'); and, for every non-empty coalition A, the epistemic operators DA (‘it is distributed knowledge among A that ...’) and CA (‘it is common knowledge among A that ...’). The formulae of CMAEL(CD) are thus defined by the following BNF expression:

\[ \varphi ::= p \mid \neg \varphi \mid (\varphi_1 \land \varphi_2) \mid \text{D}_A \varphi \mid \text{C}_A \varphi, \]

where p ranges over AP and A ranges over the set \( P^+(\Sigma) \) of non-empty subsets of Σ. The other Boolean connectives can be defined as usual. We denote formulae of CMAEL(CD) by \( \varphi, \psi, \chi, ... \) and omit parentheses in formulae whenever it does not result in ambiguity.

The distributed knowledge operator DA\( \varphi \) intuitively means that an ‘A-superagent’, who knows everything that any of the agents in A knows, can obtain \( \varphi \) as a logical consequence of their knowledge. For example, if agent a knows that \( \psi \) and agent b knows that \( \psi \rightarrow \chi \), then \( \text{D}_{\{a,b\}} \chi \) is true even though neither a nor b knows \( \chi \). The operators of individual knowledge KA\( \varphi \) (‘the agent a knows that \( \varphi \)’), for \( a \in \Sigma \), can be defined as \( \text{D}_a \varphi \), henceforth simply written \( \text{D}_a \varphi \). Then, we define

\[ \text{K}_a \varphi ::= \bigwedge_{a \in A} \text{D}_a \varphi. \]

The common knowledge operator CA\( \varphi \) intuitively means that \( \varphi \) is ‘public knowledge’ among A, i.e. that every agent in A knows that \( \varphi \) and knows that every agent in A knows that \( \varphi \), etc. Formulae of the form \( \neg \text{C}_A \varphi \) are referred to as (epistemic) eventualities, for the reasons given later on.

---

2The notion of agent used in the present article is an abstract one; in the context of distributed systems, for example, agents can be thought of as processes making up the system; in the context of multiagent systems, they can be thought of as independent software components of the system.
2.2 Coalitional multiagent epistemic models

Formulae of CMAEL(CD) are interpreted in coalitional multiagent epistemic models. In order to define those, we first need to introduce coalitional multiagent epistemic structures and frames.

**Definition 2.1**
A coalitional multiagent epistemic structure (CMAES) is a tuple

\[ \mathcal{G} = (\Sigma, S, \{R^D_A\}_{A \in \mathcal{P}(\Sigma)}, \{R^C_A\}_{A \in \mathcal{P}(\Sigma)}) \]

where
1. \( \Sigma \) is a finite, non-empty set of agents;
2. \( S \neq \emptyset \) is a set of states;
3. for every \( A \in \mathcal{P}(\Sigma) \), \( R^D_A \) is a binary relation on \( S \);
4. for every \( A \in \mathcal{P}(\Sigma) \), \( R^C_A \) is the reflexive, transitive closure of \( \bigcup_{B \subseteq A} R^D_B \).

**Definition 2.2**
A coalitional multiagent epistemic frame (CMAEF) is a CMAES

\[ \mathfrak{G} = (\Sigma, S, \{R^D_A\}_{A \in \mathcal{P}(\Sigma)}, \{R^C_A\}_{A \in \mathcal{P}(\Sigma)}) \]

where each \( R^D_A \) is an equivalence relation satisfying the following condition:

\[ (\dagger) \quad R^D_A = \bigcap_{a \in A} R^D_a \]

(Here, and further, we write \( R^D_a \) instead of \( R^D_{\{a\}} \), where \( a \in \Sigma \).)

If condition \((\dagger)\) above is replaced by the following, weaker one:

\[ (\ddagger) \quad R^D_A \subseteq R^D_B \text{ whenever } B \subseteq A, \]

then \( \mathfrak{G} \) is a coalitional multiagent epistemic pseudo-frame (pseudo-CMAEF).

Note that in every (pseudo-)CMAEF \( R^D_A \subseteq \bigcap_{a \in A} R^D_a \), and hence \( \bigcup_{B \subseteq A} R^D_B = \bigcup_{a \in A} R^D_a \). Hence, condition 4 of Definition 2.1 in (pseudo-)CMAEFs is equivalent to requiring that \( R^C_A \) is the transitive closure of \( \bigcup_{B \subseteq A} R^D_B \), for every \( A \in \mathcal{P}(\Sigma) \). Also, note that each \( R^C_A \) in a (pseudo-)CMAEF is an equivalence relation.

**Definition 2.3**
A coalitional multiagent epistemic model (CMAEM) is a tuple \( M = (\mathfrak{G}, \mathcal{AP}, L) \), where \( \mathfrak{G} \) is a CMAEF with a set of states \( S \), \( \mathcal{AP} \) is a set of atomic propositions, and \( L : S \mapsto \mathcal{P}(\mathcal{AP}) \) is a labelling function, assigning to every state \( s \) the set \( L(s) \) of atomic propositions true at \( s \).

If \( \mathfrak{G} \) is a pseudo-CMAEF, then \( M = (\mathfrak{G}, \mathcal{AP}, L) \) is a multiagent coalitional pseudo-model (pseudo-CMAEM).

The notion of truth, or satisfaction, of a CMAEL(CD)-formula at a state of a (pseudo-)CMAEM is defined in the standard Kripke semantics style. In particular:

- \( M, s \models \Box_D \phi \) iff \((s, t) \in R^D_A \) implies \( M, t \models \phi \);
- \( M, s \models \Box_C \phi \) iff \((s, t) \in R^C_A \) implies \( M, t \models \phi \).

\[^3\text{Notice that we use the same symbol, ‘\( \Sigma \)’, both for the set of names of agents in the language and for the set of agents in CMAES’s. It will always be clear from the context which set we refer to.}\]
DEFINITION 2.4
Given a (pseudo-)CMAEM $\mathcal{M}$, a CMAEL(CD)-formula $\phi$ is satisfiable in $\mathcal{M}$ if $\mathcal{M},s \models \phi$ holds for some $s \in \mathcal{M}$; $\phi$ is valid in $\mathcal{M}$ if $\mathcal{M},s \models \phi$ holds for every $s \in \mathcal{M}$.

A formula $\phi$ is satisfiable if it is satisfiable in some CMAEM; it is valid, denoted $\models \phi$, if it is valid in every CMAEM.

The satisfaction condition for the operator $C_A$ can be re-stated in terms of reachability. Let $\mathcal{M}$ be a (pseudo-)CMAEM with state space $S$ and let $s,t \in S$. We say that $t$ is $A$-reachable from $s$ if either $s=t$ or, for some $n \geq 1$, there exists a sequence $s=s_0,s_1,\ldots,s_n=t$ of elements of $S$ such that, for every $0 \leq i < n$, there exists $a_i \in A$ such that $(s_i,s_{i+1}) \in R_D^{a_i}$. It is then easy to see that the satisfaction condition for $C_A$ is equivalent to the following one:

- $\mathcal{M},s \models C_A \phi$ iff $\mathcal{M},t \models \phi$ for every $t$ that is $A$-reachable from $s$.

The following claim be easily verified.

**Proposition 2.5**
$\models C_A \phi \iff \phi \land \bigwedge_{a \in A} D_a C_A \phi$.

**Remark** If $\Sigma = \{a\}$, then $D_a \phi \iff C_a \phi$ is valid for all $\phi$. Thus, the single-agent case is essentially trivialized and, therefore, we assume throughout the remainder of the article that the set $\Sigma$ of names of agents in the language of CMAEL(CD) contains at least 2 agents.

### 3 Hintikka structures for CMAEL(CD)

We are ultimately interested in (constructive) satisfiability of (finite sets of) formulae in models. However, the tableau procedure we present in this article checks for the existence of a more general kind of semantic structure for the input formula, namely a Hintikka structure. In Section 3.1 we introduce Hintikka structures for CMAEL(CD). In Section 3.2 we show that satisfiability in Hintikka structures is equivalent to satisfiability in models; consequently, testing for satisfiability in a Hintikka structure can replace testing for satisfiability in a model.

#### 3.1 Fully expanded sets and Hintikka structures

There are two fundamental differences between (pseudo-)models and Hintikka structures for CMAEL(CD), which make working with the latter substantially easier than working directly with models. First, while models specify the truth value of every formula of the language at each state, Hintikka structures only do so for the formulae relevant to the evaluation of a fixed formula $\theta$ (or, a finite set of formulae $\Theta$) at a distinguished state. Second, the relations in (pseudo-) models have to satisfy certain conditions (see Definition 2.2), while in Hintikka structures conditions are only placed on the labels of states. These labelling conditions ensure, however, that every Hintikka structure generates, through the constructions described in Section 3.2, a pseudo-model so that membership of formulae in the labels is compliant with the truth in the resultant pseudo-model. We then show how to convert a pseudo-model into a bona fide model in a ‘truth-preserving’ way.

To describe Hintikka structures, we need the concept of fully expanded set. Such sets contain all the formulae that have to be satisfied locally at the state under consideration. We divide all the formulae that are not elementary in the sense that their satisfaction at the state does not imply satisfaction of any other formulae at the same state (such as $p \in AP$ or $\neg D_A \phi$) into $\alpha$-formulae and $\beta$-formulae. The former are formulae of a conjunctive type, i.e. their truth implies the truth of all
Tableaux for the full coalitional multiagent epistemic logic

Table 1. $\alpha$- and $\beta$-formulas of CMAEL(CD) with their respective components

<table>
<thead>
<tr>
<th>$\alpha$-formula</th>
<th>$\alpha$-components</th>
<th>$\beta$-formula</th>
<th>$\beta$-components</th>
</tr>
</thead>
</table>
| $\neg
\neg\varphi$   | $\{\varphi\}$      | $\neg(\varphi \land \psi)$ | $\{\neg\varphi, \neg\psi\}$ |
| $\varphi \land \psi$ | $\{\varphi, \psi\}$ | $\neg C_A \varphi$ | $\{\neg\varphi\} \cup \{\neg D_a C_A \varphi | a \in A\}$ |
| $D_A \varphi$    | $\{D_A \varphi, \varphi\}$ |                 |                   |
| $C_A \varphi$    | $\varphi \cup \{D_a C_A \varphi | a \in A\}$ |                 |                   |

their $\alpha$-components at the same state, while the latter are of a disjunctive type: their truth implies the truth of at least one of their $\beta$-components at the same state. Table 1 shows the $\alpha$- and $\beta$-formulas of CMAEL(CD) together with their $\alpha$- and $\beta$-components. The following claims are straightforward, the cases of common knowledge using Proposition 2.5.

**Lemma 3.1**

1. Every $\alpha$-formula is equivalent to the conjunction of its $\alpha$-components.
2. Every $\beta$-formula is equivalent to the disjunction of its $\beta$-components.

**Definition 3.2**

The closure of the formula $\varphi$ is the smallest set of formulae $\cl(\varphi)$ such that:

1. $\varphi \in \cl(\varphi)$;
2. $\cl(\varphi)$ is closed with respect to $\alpha$- and $\beta$-components of all $\alpha$- and $\beta$-formulae, respectively;
3. for any formula $\psi$ and coalition $A$, if $\neg D_A \psi \in \cl(\varphi)$ then $\neg \psi \in \cl(\varphi)$.

**Definition 3.3**

For any set of formulae $\Delta$ we define $\cl(\Delta) := \bigcup \{\cl(\varphi) | \varphi \in \Delta\}$. A set of formulae $\Delta$ is closed if $\Delta = \cl(\Delta)$.

**Remark 3.4**

Intuitively, the closure of a set of formulae $\Gamma$ consists of all formulae that may appear in the tableau whose input is the set of formulae $\Gamma$.

**Definition 3.5**

A set of formulae is patently inconsistent if it contains a contradictory pair of formulae $\varphi$ and $\neg \varphi$.

**Definition 3.6**

A set $\Delta$ of CMAEL(CD)-formulae is fully expanded if it satisfies the following conditions:

- $\Delta$ is not patently inconsistent;
- if $\varphi$ is an $\alpha$-formula and $\varphi \in \Delta$, then all $\alpha$-components of $\varphi$ are in $\Delta$.
- if $\varphi$ is a $\beta$-formula and $\varphi \in \Delta$, then at least one $\beta$-component of $\varphi$ is in $\Delta$.

Intuitively, a non-patently inconsistent set is fully expanded if it is closed under applications of all local (pertaining to the same state of a structure) formula decomposition rules.

**Definition 3.7**

The procedure FullExpansion applies to a set of formulae $\Gamma$ and produces a (possibly empty) family of sets $\mathcal{FE}(\Gamma)$, called the family of full expansions of $\Gamma$, obtained as follows: start with the singleton family $\{\Gamma\}$; if $\Gamma$ is patently inconsistent, halt and return $\mathcal{FE}(\Gamma) = \emptyset$; otherwise repeatedly apply, until saturation, the following set replacement operations, each time to a non-deterministically chosen...
Tableaux for the full coalitional multiagent epistemic logic

Set $\Phi$ from the current family of sets $\mathcal{F}$ and a formula $\varphi \in \Phi$; though, we prioritize the eventualities in $\Gamma$ so that these formulae are processed first:

1. If $\varphi$ is an $\alpha$-formula with $\alpha$-components $\varphi_1$ and $\varphi_2$, then replace $\Phi$ by $\Phi \cup \{\varphi_1, \varphi_2\}$.
2. If $\varphi$ is a $\beta$-formula such that none of its $\beta$-components is in $\Phi$, then replace $\Phi$ with the family of extensions

$$\{\Phi \cup \{\psi\} \mid \psi \text{ is a $\beta$-component of } \varphi\}$$

3. If $\varphi = \neg C_A \psi$ and $\neg \psi \notin \Phi$, but some of the other $\beta$-components of $\varphi$ is in $\Phi$, then add to $\mathcal{F}$ the set $\Phi \cup \{\neg \psi\}$.

The following proviso applies to the procedure above: if a patently inconsistent set is added to $\mathcal{F}$ as a result of such application, it is removed immediately thereafter.

Saturation occurs when no application of a set replacement operation can change the current family $\mathcal{F}$. At that stage, the family $\mathcal{F}(\Gamma)$ of sets of formulae is produced and returned. Reaching a stage of saturation is guaranteed to occur because all sets of formulae produced during the procedure FullExpansion are subsets of the finite set $\mathcal{E}(\Gamma)$.

Notice that the procedure FullExpansion allows adding not more than one $\beta$-component of a formula $\varphi = \neg C_A \psi$ to the initial set, besides $\neg \psi$.

In what follows, we will need the following proposition.

**Proposition 3.8**

For any finite set of formulae $\Gamma$:

$$\models \bigwedge \Gamma \iff \bigvee \left\{ \bigwedge \Delta \mid \Delta \in \mathcal{F}(\Gamma) \right\}.$$  

**Proof.** By Lemma 3.1, every set replacement operation applied to a family $\mathcal{F}$ preserves the formula $\bigvee \left\{ \bigwedge \Delta \mid \Delta \in \mathcal{F}(\Gamma) \right\}$ up to logical equivalence. At the beginning, that formula is $\bigwedge \Gamma$, hence the claim follows. \qed

We now define Hintikka structures for CMAEL(CD):

**Definition 3.9**

A coalitional multiagent epistemic Hintikka structure (CMAEHS) is a tuple

$$(\Sigma, S, \{\mathcal{R}_A^0\}_{A \in \mathcal{P}^+(\Sigma)}, \{\mathcal{R}_A^C\}_{A \in \mathcal{P}^+(\Sigma)}, \mathcal{AP}, H)$$

such that:

* $(\Sigma, S, \{\mathcal{R}_A^0\}_{A \in \mathcal{P}^+(\Sigma)}, \{\mathcal{R}_A^C\}_{A \in \mathcal{P}^+(\Sigma)})$ is a CMAES (recall Definition 2.1);
* $\mathcal{AP}$ is a set of atomic propositions;
* $H$ is a labelling of the elements of $S$ with sets of CMAEL(CD)-formulae that satisfy the following constraints, for every $s, s' \in S$:

**CH1** $H(s)$ is fully expanded;
**CH2** If $\neg D_B \varphi \in H(s)$, then $(s, t) \in \mathcal{R}_B^0$ and $\varphi \in H(t)$, for some $t \in S$;
**CH3** If $(s, s') \in \mathcal{R}_A^C$, then $D_B \varphi \in H(s)$ iff $D_B \varphi \in H(s')$, for every $B \subseteq A$;
**CH4** If $\neg C_A \varphi \in H(s)$, then $(s, t) \in \mathcal{R}_A^C$ and $\neg \varphi \in H(t)$, for some $t \in S$.  


3.2 Equivalence of Hintikka structures and models for CMAEL(CD)

Here we show that satisfiability in Hintikka structures is equivalent to satisfiability in models. For brevity, we only deal with single formulae; the extension to finite sets of formulae is straightforward. The main complications in the proofs below arise due to the presence of distributed knowledge operators in the language of a logic.

Here we will prove that a CMAEL(CD)-formula \( \theta \) is satisfiable in a CMAEM iff it is satisfiable in a CMAEHS. First, we show that satisfiability in a CMAEM implies satisfiability in a CMAEHS. Then, we show that satisfiability in a CMAEHS implies satisfiability in a pseudo-CMAEM, which in turn implies satisfiability in a CMAEM.

That satisfiability in a CMAEM implies satisfiability in a CMAEHS is almost immediate. Given a CMAEM \( \mathcal{M} \) with a set of states \( S \), define the extended labelling function \( L_{\mathcal{M}}^+ \) from \( S \) to the power-set of \( \text{CMAEL(CD)} \)-formulae as follows: \( L_{\mathcal{M}}^+(s) = \{ \varphi | \mathcal{M}, s \models \varphi \} \). It is then routine to check the following.

**Lemma 3.11**

Let \( \mathcal{M} = (\Sigma, S, (R^D_A)_{A \in \mathcal{P}^+(\Sigma)}, (R^C_A)_{A \in \mathcal{P}^+(\Sigma)}, \mathcal{AP}, L) \) be a CMAEM satisfying \( \theta \) and let \( L_{\mathcal{M}}^+ \) be the extended labelling on \( \mathcal{M} \). Then, \( (\Sigma, S, (R^D_A)_{A \in \mathcal{P}^+(\Sigma)}, (R^C_A)_{A \in \mathcal{P}^+(\Sigma)}, \mathcal{AP}, L_{\mathcal{M}}^+) \) is a CMAEHS satisfying \( \theta \). Therefore, satisfiability in a CMAEM implies satisfiability in a CMAEHS.

For the converse direction we need two steps, done in Lemma 3.12 and Lemma 3.14.

**Lemma 3.12**

Let \( \theta \) be a CMAEL(CD)-formula satisfiable in a CMAEHS. Then, \( \theta \) is satisfiable in a pseudo-CMAEM.

**Proof.** Let \( \mathcal{H} = (\Sigma, S, (R^D_A)_{A \in \mathcal{P}^+(\Sigma)}, (R^C_A)_{A \in \mathcal{P}^+(\Sigma)}, \mathcal{AP}, H) \) be a CMAEHS for \( \theta \). We construct a pseudo-CMAEM \( \mathcal{M} \) satisfying \( \theta \) out of \( \mathcal{H} \) as follows.

First, for every \( A \in \mathcal{P}^+(\Sigma) \), let \( R^D_A \) be the reflexive, symmetric and transitive closure of \( \bigcup_{s \in \theta} R^D_A \) and let \( R^C_A \) be the transitive closure of \( \bigcup_{s \in \theta} R^C_A \). Thus, both \( R^D_A \) and \( R^C_A \) are equivalence relations and \( R^D_A \subseteq R^D_{A'} \) and \( R^C_A \subseteq R^C_{A'} \), for every \( A \in \mathcal{P}^+(\Sigma) \). Second, let \( L(s) = H(s) \cap \mathcal{AP} \), for every \( s \in S \). It is then immediate to check that \( B \subseteq A \) implies \( R^D_B \subseteq R^D_A \), and hence, \( \mathcal{M}' = (\Sigma, S, (R^D_A)_{A \in \mathcal{P}^+(\Sigma)}, (R^C_A)_{A \in \mathcal{P}^+(\Sigma)}, \mathcal{AP}, L) \) is a pseudo-CMAEM.

Basically this construction relabels the edges of a Hintikka structure such that if a directed edge is labelled with a coalition \( A \), it is made bidirectional and is further labelled with all coalitions that are subsets of \( A \). Hereafter the relation is then made transitive and reflexive. The labels of the states are reduced to only containing (positive) atoms. Figure 1 illustrates the process of transforming the Hintikka structure on the left into the pseudo-model on the right.

To complete the proof of the lemma, we show, by induction on the structure of the formulae in \( \Theta(\theta) \) that, for every \( s \in S \) and every formula \( \chi \), the following hold:

1. \( \chi \in H(s) \) implies \( \mathcal{M}', s \models \chi \);
2. \( \neg \chi \in H(s) \) implies \( \mathcal{M}', s \models \neg \chi \).

The statement of the lemma then follows.
DEFINITION 3.13

Let \( \mathcal{M} = (\Sigma, S, \{ R_i \}_{i \in \mathbb{N}^+}, \mathbb{N}, \mathbb{A}, L) \) be a (pseudo-) CMAEM and let \( s, t \in S \). A maximal path from \( s \) to \( t \) in \( \mathcal{M} \) is a sequence \( s_0, A_0, s_1, A_1, \ldots, s_{n-1}, A_{n-1}, s_n \) where \( s = s_0 \) and \( t = s_n \), such that \( n = 0 \) and \( s = t \) or, for every \( 0 \leq i < n \), \( (s_i, s_{i+1}) \in R_i \), but \((s_i, s_{i+1}) \in R_i \) does not hold for any \( B \) with \( A_i \subset B \subset \Sigma \). A segment \( \rho' \) of a maximal path \( \rho \) starting and ending with a state is a sub-path of \( \rho \).
LEMMA 3.14
Let $\theta$ be a CMAEL(CD)-formula satisfiable in a pseudo-CMAEM; then, $\theta$ is satisfiable in a CMAEM.

PROOF. Suppose that $\theta$ is satisfied in a pseudo-CMAEM $M$ at state $s$. Let $M_s = (\Sigma, S, [R^0_A]_{A \in \mathcal{P}^i(\Sigma)}, \{R^0_A\}_{A \in \mathcal{P}^i(\Sigma)}, \mathcal{AP}, L)$ be the submodel of $M$ generated by $s$. Then, $M_s \vDash \theta$ since $M_s$ and $M$ are locally bisimilar at $s$. Next, we unravel $M_s$ into a model $M^* = (\Sigma, S^*, [R^0_A]_{A \in \mathcal{P}^i(\Sigma)}, \{R^0_A\}_{A \in \mathcal{P}^i(\Sigma)}, \mathcal{AP}, L^*)$, as follows.

First, call a maximal path $\rho$ in $M_s$ an $s$-max-path if the first component of $\rho$ is $s$, and let $S^*$ be the set of all $s$-max-paths in $M_s$. Notice that $s$ by itself is an $s$-max-path with $l(s) = s$.

For every $A \in \mathcal{P}^i(\Sigma)$, let

$$R^0_A = \{ (\rho, \tau) \mid \rho, \tau \in S^*, \tau_{|\tau| - 1} = \rho \text{ and } l(\tau) \subseteq A \},$$

i.e. $(\rho, \tau) \in R^0_A$ if $\tau$ extends $\rho$ with one step labelled by a coalition containing $A$. Next, let $R^*_{A,B}$ be a reflexive, symmetric and transitive closure of $R^0_A$. Notice that $(\rho, \tau) \in R^*_{A,B}$ holds for two distinct paths $\rho$ and $\tau$ iff there exists a sequence $\rho_0, \ldots, \rho_n \in S^*$ with $\rho = \rho_0$ and $\tau = \rho_n$ such that for all $i < n$, either $(\rho_i, \rho_{i+1}) \in R^*_{A,B}$ or $(\rho_{i+1}, \rho_i) \in R^*_{A,B}$. It then follows that the following downward closure condition holds:

**DC** If $(\rho, \tau) \in R^*_{A,B}$ and $B \subseteq A$, then $(\rho, \tau) \in R^*_{A,B}$.

The relations $R^*_{A,B}$ are defined as in any CMAEF. To complete the definition of $M^*$, we put $L^*(\rho) = L(l(\rho))$, for every $\rho \in S^*$. Notice that $M^*$ is tree-like in the sense that the structure $(S^*, [R^*_{A,B}]_{A \in \mathcal{P}^i(\Sigma)})$ is a tree.

By this construction we basically remove all ‘non-maximal’ edges between two vertices from the part of the given pseudomodel that can be reached by the given state $s$. Then we build paths by starting in $s$ and then traversing the resulting graph via the edges. E.g. if we consider the pseudomodel $M$ in Figure[1] and we let the top-left-most state be $s$, then $M_s \vDash \neg D_s \neg p \land D_b q$. $S^*$ will in this case be all paths starting in $s$ and following the links in the graph.

\[
\begin{array}{c}
\{a,b\} \quad s \quad \{a,b\} \\
\{a\} \quad \{a\} \quad \{a\} \quad \{a\} \\
\{a\} \quad \{a\} \quad \{a\} \quad \{a\} \\
\{a\} \quad \{a\} \quad \{a\} \quad \{a\} \\
\end{array}
\]

I.e. $\rho = (s, \{a, b\}, s, \{a\}, t)$ and $\tau = (s, \{a, b\}, r, \{b\}, t)$ are in $S^*$, while $\rho' = (s, \{a\}, s, \{b\}, t) \notin S^*$.

We have $(\rho, \tau) \notin R^*_{a,b}$, $(\rho, \tau) \notin R^*_{b,b}$ and $(\rho, \tau) \notin R^*_{a,b}$. On the other hand, $(\tau, (s, \{a, b\}, r, \{a, b\}, s)) \in R^*_{a,b}.

In this example, $L^*(\rho) = L^*(\rho) \overset{\text{def}}{=} L(t) = \{p, q\}$.
Tableaux for the full coalitional multiagent epistemic logic

FIG. 2. The situation from (3.1) drawn in $M$, i.e. the dots/circles belong to $S$, and the links are links in $R^D$.

It is clear from the construction, namely from (DC), that $M^*$ is a pseudo-CMAEM, and in the following, we will show that condition (†) of Definition 2.2 also holds, so that $M^*$ is a CMAEM.

First, we notice that, since $M^*$ is tree-like, we have $(\rho, \tau) \in R^D_A$ iff there exists $k \geq 0$, with $k \leq |\rho|$ and $k \leq |\tau|$, such that

$$\rho|_k = \tau|_k,$$

and for all $k < i \leq |\tau|$ and $k < j \leq |\rho|$, $A \subseteq sl(\tau|_i)$ and $A \subseteq sl(\rho|_j)$. (3.1)

(The situation is depicted in Figure 2) As stated, we have to prove that $R^D_A = \bigcap_{a \in A} R^D_a$ for every $A \in \mathcal{P}^+(\Sigma)$. The left-to-right inclusion immediately follows from (DC). For the converse, assume that $(\rho, \tau) \in R^D_A$ holds for every $a \in A$. Then, for every $a \in A$, according to (3.1), there exists $k_a \geq 0$ such that $\rho|_{k_a} = \tau|_{k_a}$ and $[a] \subseteq sl(\tau|_{k_a})$, $sl(\rho|_{k_a})$ for every $|\tau| \geq i > k_a$ and every $|\rho| \geq j > k_a$. Now, let $k$ be the largest $k_a$ satisfying this condition (such a $k$ exists since $M^*$ is tree-like). Then, $\rho_k = \tau_k$, and for every $a \in A$, the inclusions $[a] \subseteq sl(\tau|_i)$ and $[a] \subseteq sl(\rho|_j)$ hold for every $|\tau| \geq i > k$ and every $|\rho| \geq j > k$. Therefore, condition (3.1) is fulfilled for $A$ and $k$, and hence $(\rho, \tau) \in R^D_A$, as desired.

Finally, it remains to prove that $M^*$ satisfies $\theta$. From (3.1) we see, that if $(\rho, \tau) \in R^D_A$, then $(l(\rho), l(\tau)) \in R^D_A$, since every $R^D_A$ is an equivalence relation. It is now easy to check that the relation $Z = \{(\rho, l(\rho)) | \rho \in S^*\}$ is a bisimulation between $M^*$ and $M_a$. Since $(s, l(s)) \in Z$, it follows that $M^*, s \models \theta$, and we are done.

**Theorem 3.15**
Let $\theta$ be a CMAEL(CD)-formula. Then, $\theta$ is satisfiable in a CMAEHS iff it is satisfiable in a CMAEM.

**Proof.** Immediate from Lemmas 3.11, 3.12 and 3.14.

4 Tableau procedure for testing satisfiability in CMAEL(CD)

In this section, we present our tableau algorithm for checking (constructive) satisfiability of formulae of CMAEL(CD). We start off by explaining the general philosophy underlying our tableau procedure and then present it in detail.

4.1 Basic ideas and overview of the tableau procedure

Traditionally, the propositional tableau method works by decomposing the formula whose satisfiability is being tested into its $\alpha$-, resp. $\beta$- components—repeatedly, until producing all full expansions of that formula. All these components belong to the closure of the input formula. When the closure is
finite (as it is usually the case with modal and temporal logics) the termination of the tableau-building procedure is guaranteed because there are only finitely many full expansions.

Furthermore, in the tableau method for the classical propositional logic that decomposition into components produces a tree representing an exhaustive search for a Hintikka set, the propositional analogue of Hintikka structures, for the input formula. If at least one branch of the tree remains open, it produces a full expansion of the input formula, which is a Hintikka set for this formula. In this case, the formula is pronounced satisfiable; otherwise, it is declared unsatisfiable. In the case of modal and temporal logics, local decomposition steps, producing full expansions, are interleaved with steps along the accessibility/transition relations, producing sets of formulae that are supposed to be true at successors of the current state. These sets are subjected, again, to local decomposition into components, eventually producing their full expansions, etc. In order to distinguish fully expanded sets from those produced after transition to successors, we will deal with two types of nodes of the tableau, respectively called ‘states’ and ‘prestates’. In order to ensure termination of the construction process, we will systematically reuse states and prestates labelled with the same sets of formulae.

The tableau procedure for testing a formula \( \theta \) for satisfiability attempts to construct a non-empty graph \( T^\theta \) (called itself a tableau) representing ‘sufficiently many’ CMAEHSs for \( \theta \) in the sense that if \( \theta \) is satisfiable in any CMAEHS, then it is satisfiable in a CMAEHS represented by the tableau. The procedure consists of three major sub-procedures, or phases: construction, prestate elimination and state elimination. During the construction phase, we build the pretableau \( P^\theta \)—a directed graph whose nodes are sets of formulae of two types: states and prestates, as explained above. States represent (labels of) states of the CMAEHSs that the tableau attempts to construct, while prestates are only used temporarily, during the construction phase.

During the prestate elimination phase, we create a smaller graph \( T_0^\theta \) out of \( P^\theta \), called the initial tableau for \( \theta \), by eliminating all the prestates of \( P^\theta \) and adjusting its edges, as prestates have already fulfilled their role of keeping the graph finite and can, therefore, be discharged.

In the case of classical propositional logic, the only reason why it may 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 a patent inconsistency. In the case of logics with fixpoint-definable operators, such as CMAEL(CD), there are two other reasons for a tableau not to correspond to any Hintikka structure for the input formula. The first one has to do with realization of eventualities—formulas of the form \( \neg C_A \phi \), whose truth condition requires that \( \neg \phi \) ‘eventually’ becomes true—in the tableau graph. Applying decomposition rules to eventualities in the construction of the tableau can postpone indefinitely the realization by keeping ‘promising’ that the realization will happen further down the line, while that ‘promise’ never becomes fulfilled. Therefore, a ‘good’ tableau should not contain states with unrealized eventualities. The other additional reason for the resultant tableau not to represent a Hintikka structure is that some states do not have all the successors they would be required to have in a corresponding Hintikka structure (for example, because those successors have been removed for not realizing eventualities).

During the state elimination phase, we remove from \( T_0^\theta \) all states, if any, that cannot be satisfied in any CMAEHS for any of the reasons suggested above and discussed in more detail further (excluding patently inconsistent sets, which are removed ‘on the fly’ during the construction phase). The elimination procedure results in a (possibly empty) subgraph \( T^\theta \) of \( T_0^\theta \), called the final tableau for \( \theta \). If some state \( \Delta \) of \( T^\theta \) contains \( \theta \), it is declared satisfiable; otherwise, \( \theta \) is declared unsatisfiable.

\[\text{From now on we will use the term ‘state’ in two related but distinct senses: as a state of a tableau and as a state of a semantic structure (frame, model, Hintikka structure). The use of term ‘state’ will usually be clear from the context or explicitly specified.}\]
The logic $\text{CMAEL(CD)}$ involves modal operators over equivalence relations, and thus invokes some typical complications in the tableau-building procedures associated with inverse-looking modalities, see e.g. [19]: every box occurring in the label of a descendant state has a backwards effect on all predecessor states, incl. the current state. In order to deal with these complications we must either organize a mechanism for backtracking and backwards propagation of box-formulae, or a mechanism for anticipation of the occurrence of such boxes in the future, coming from subformulae of formulae in the label of the current state, based on analytic cut rules. We will adopt here the latter approach, which is easier to describe and implement into what we call a diamond-propagating procedure, by employing suitably restricted analytic cut rules to maintain the efficiency of the procedure, but later we will briefly discuss the former alternative, too. The two procedures only differ in the construction phase; the prestate and state elimination phases are common to both. The need and use of analytic cut rules is illustrated later in Example 5.6.

4.2 Cut-saturated sets and expansions

The application of the analytic cut, mentioned above, is implemented by imposing an additional cut-saturating rule on the construction of the full expansions of a given set of formulae. In order to prevent the unnecessary swelling and proliferation of states, we will restrict the application of that rule by imposing generic restrictions which, on the other hand, should be sufficiently relaxed to guarantee the completeness of the tableau procedure. These generic conditions, which will be specified later, will be imposed separately on the two types of box-formulae in $\text{CMAEL(CD)}$, viz. $D_A$-formulae and on $C_A$-formulae.

**Definition 4.1**
Given restrictive conditions $C_1$ and $C_2$, a set $\Delta$ of $\text{CMAEL(CD)}$-formulae is $(C_1,C_2)$-cut-saturated if it satisfies the following conditions, where $\text{Sub}(\psi)$ is the set of subformulae of a formula $\psi$:

- **CS0** $\Delta$ is fully expanded (recall Definition 3.6).
- **CS1** For any $D_A\phi \in \text{Sub}(\psi)$ where $\psi \in \Delta$, if condition $C_1$ holds then either $D_A\phi \in \Delta$ or $\neg D_A\phi \in \Delta$.
- **CS2** For any $C_A\phi \in \text{Sub}(\psi)$ where $\psi \in \Delta$, if condition $C_2$ holds then either $C_A\phi \in \Delta$ or $\neg C_A\phi \in \Delta$.

We note that [CS1] and [CS2] are semantically sound rules, no matter what $C_1$ and $C_2$ are, as they cannot make a tableau closed if the input formula is satisfiable. On the other hand, if $C_1$ and $C_2$ are too strong, that may prevent the tableau from closing and thus yield an incomplete tableau procedure, as will become apparent later. Again, the reason we would want to make $C_1$ and $C_2$ as strong as possible is to avoid branching on too many formulae, causing an unnecessary large state space and resulting in a practically less efficient procedure.

Hereafter, we will omit the explicit mention of the conditions $C_1$ and $C_2$, unless necessary. In fact, for now we can assume both $C_1$ and $C_2$ to be True, but later we will introduce non-trivial restrictive conditions.

**Definition 4.2**
The family $\text{CSE}(\Gamma)$ of cut-saturated expansions (CS-expansions) of a set of formulae $\Gamma$ is defined by expanding the procedure $\text{FULL Expansion}$ with the following two set-replacement rules, again applied to a non-deterministically chosen set $\Phi$ from the current family and a formula $\psi \in \Phi$:

1. For any formula $D_A\phi$ that is a subformula of $\psi$ such that $C_1$ is satisfied, replace $\Phi$ with the two extensions of $\Phi$ obtained by adding respectively $D_A\phi$ and $\neg D_A\phi$ to it.
2. For any formula $C_A\phi$ that is a subformula of $\psi$ such that $C_2$ is satisfied, replace $\Phi$ with the two extensions of $\Phi$ obtained by adding respectively $C_A\phi$ and $\neg C_A\phi$ to it.
It is clear from the definition that all sets in $CSE(\Gamma)$ are $(C_1, C_2)$-cut-saturated.

**Definition 4.3**

The extended closure of $\theta$, denoted $\text{ecl}(\theta)$, is the smallest set such that $\varphi, \neg \varphi \in \text{ecl}(\theta)$ for every $\varphi \in \text{cl}(\theta)$. The extended closure $\text{ecl}(\Gamma)$ of a set of formulae $\Gamma$ is defined likewise.

The following is immediate from the definitions.

**Lemma 4.4**

Every CS-expansion of a set of formulae $\Gamma$ is a subset of $\text{ecl}(\Gamma)$.

**Lemma 4.5**

For any CMAEL(CD)-formula $\theta$, the size of (i.e. number of formulae in) the extended closure of $\theta$ is $O(k \cdot |\theta|)$, where $k$ is the number of agents occurring in $\theta$.

**Proof.** Straightforward.

### 4.2.1 Construction phase

As already mentioned, a tableau algorithm attempts to produce a compact representation of ‘sufficiently many’ CMAEHSs for the input formula; in this attempt, it sets in motion an exhaustive search for such CMAEHSs. As a result, the pretableau $P^n$ built at this phase contains two types of edge, as well as two types of node (states and prestates; see above).

One type of edge, depicted by unmarked, dashed uni-directed arrows $\rightarrow$, represents the search dimension of the tableaux. The exhaustive search considers all possible alternatives arising when prestates are expanded into states by branching in the ‘disjunctive’ cases. Thus, when we draw unmarked arrows from a prestate $\Gamma$ to each state from a set of states $X$, this intuitively means that, in any CMAEHS, a state satisfying $\Gamma$ has to satisfy at least one of the states in $X$.

The second type of edge represents transition relations in the CMAEHSs that the procedure attempts to build. Accordingly, this type of edges is represented by solid, uni-directed arrows $\rightarrow$, marked with formulae whose presence in one of the end nodes requires the presence in the tableau of the other end node, reachable by a particular relation. Intuitively, if $\neg D_{\Delta} \varphi \in \Delta$ for some state $\Delta$, then some (state obtained from a) prestate $\Gamma$ containing $\neg \varphi$ must be accessible from $\Delta$ by relation $R^\nu_\Delta$. We mark these arrows with the respective formulae $\neg D_{\Delta} \varphi$ in order to keep track of the specific reason for creating that particular state. That information will be needed during the elimination phases.

We now turn to presenting the rules of the ‘diamond-propagating’ construction phase, each of which creates a different type of edge, as discussed above. The first rule, (SR), prescribes how to create states from prestates, while (DR) expands prestates into states.

**Rule (SR)**

Given a prestate $\Gamma$, such that (SR) has not been applied to it before, do the following:

1. Add to the pretableau all CS-expansions $\Delta$ of $\Gamma$; declare these to be states;
2. For each so obtained state $\Delta$, put $\Gamma \rightarrow \Delta$;
3. If, however, the pretableau already contains a state $\Delta' = \Delta$, then do not create a new state, but put $\Gamma \rightarrow \Delta'$.

We denote by states$(\Gamma)$ the (finite) set $\{\Delta \mid \Gamma \rightarrow \Delta\}$.

**Rule (DR)**

Given a state $\Delta$ such that $\neg D_{\Delta} \varphi \in \Delta$ and (DR) has not been applied to $\Delta$ with respect to $\neg D_{\Delta} \varphi$ before, do the following:
Tableaux for the full coalitional multiagent epistemic logic

1. Add to the pretableau the set \( \Gamma_1 = \{ \neg \phi \} \cup \{ D_{A'} \psi \in \Delta \mid A' \subseteq A \} \cup \{ \neg D_{A'} \psi \neq \neg D_{A} \phi \} \cup \{ \neg C_{A'} \psi \in \Delta \mid A' \cap A \neq \emptyset \} \) and declare this set to be a prestate.

2. Put \( \Delta \rightarrow \Gamma_1 \).

3. If, however, the pretableau already contains a prestate \( \Gamma_1' = \Gamma_1 \), then do not create a new prestate, but put \( \Delta \rightarrow \Gamma_1' \).

When building a tableau for a formula \( \theta \), the construction phase begins with creating a single prestate \( \{ \theta \} \). Afterwards, we alternate between (SR) and (DR): first, (SR) is applied to the prestates created at the previous stage of the construction, then (DR) is applied to the states created at the previous stage.

The construction phase is completed when every prestate required to be added to the pretableau has already been added (as prescribed in item 3 of (SR)) and (DR) does not apply to any of the states with respect to any of the formulae.

**Example 4.6**
Let us construct the pretableau for the formula \( \theta = \neg D_{a,c} C_{a,b} p \land C_{a,b} (p \land q) \), assuming that \( \Sigma = \{ a, b, c \} \). To save space, we replace \( \theta \) by the set of its conjuncts \( \Theta = \{ \neg D_{a,c} C_{a,b} p, C_{a,b} (p \land q) \} \).

Here and further on in the examples, we let \( C_A \phi \) denote the set \( \{ C_A \phi, \phi \} \cup \bigcup_{a \in A} D_a C_A \phi \). Figure 3 shows the pretableau for \( \Theta \).

### 4.2.2 Prestate elimination phase
At this phase, we remove from pretableau \( T_\theta \) all the prestates and unmarked arrows, by applying the following rule (the resultant graph is denoted \( T_\theta^0 \) and is called the initial tableau):

**(PR)** For every prestate \( \Gamma \) in \( T_\theta \), do the following:

1. Remove \( \Gamma \) from \( T_\theta \);
2. If there is a state \( \Delta \) in \( T_\theta \) with \( \Delta \rightarrow \Gamma \), then for every state \( \Delta' \in \text{states}(\Gamma) \), put \( \Delta \rightarrow \Delta' \);
Tableaux for the full coalitional multiagent epistemic logic

4.2.3 State elimination phase

During this phase, we remove from $T_0^\phi$ states that are not satisfiable in any CMAEHS. Of course, when a state is removed, so are all of its incoming and outgoing arrows.

There are two reasons why a state $\Delta_1$ of $T_0^\phi$ might turn out to be unsatisfiable: either because $\Delta_1$ needs, in order to satisfy some diamond-formula, a successor state that has been eliminated, or because $\Delta_1$ contains an eventuality that is not realized in the tableau. Accordingly, we have two elimination rules (E1) and (E2).

Formally, the state elimination phase is divided into stages; we start at stage 0 with $T_0^\phi$; at stage $n + 1$, we remove from the tableau $T_n^\phi$ obtained at the previous stage exactly one state, by applying one of the elimination rules, thus obtaining the tableau $T_{n+1}^\phi$. We state the rules below, where $S_n^\phi$ denotes the set of states of $T_n^\phi$.

(E1) If $\Delta_1 \in S_n^\phi$ contains a formula $\chi = \neg D_a \psi$ such that there is no $\Delta_1 \chi \rightarrow \Delta_1'$, where $\Delta_1' \in S_n^\phi$, then obtain $T_{n+1}^\phi$ by eliminating $\Delta_1$ from $T_n^\phi$.

For the other elimination rule, we need the concept of eventuality realization.

Definition 4.8

The eventuality $\xi = \neg C_A \psi$ is realized at $\Delta$ in $T_n^\phi$ if either $\neg \psi \in \Delta$ or there exists in $T_n^\phi$ a finite number of states $\Delta_0, \Delta_1, \ldots, \Delta_m$ such that $\Delta_0 = \Delta; \neg \psi \in \Delta_m$; and, for every $0 \leq i < m$, $\xi \in \Delta_i$ and there exists $\chi_i = \neg D_a \psi_i$ such that $a_i \in A$ and $\Delta_i \chi_i \rightarrow \Delta_{i+1}$.

We can now state the rule.

(E2) If $\Delta \in S_n^\phi$ contains an eventuality $\neg C_A \psi$ that is not realized at $\Delta$ in $T_n^\phi$, then obtain $T_{n+1}^\phi$ by removing $\Delta$ from $T_n^\phi$.

We check for realization of $\neg C_A \psi$ by running the following, global procedure that marks all states of $T_n^\phi$ realizing $\neg C_A \psi$ in $T_n^\phi$. Initially, we mark all $\Delta \in S_n^\phi$ such that $\neg \psi \in \Delta$. Then, we repeatedly do the following: if $\Delta \in S_n^\phi$ contains $\neg C_A \psi$ and is unmarked yet, but there exists at least one $\Delta'$ such that $\Delta \rightarrow \Delta'$, for some formula $\psi$ and $a \in A$ and $\Delta'$ is marked, we mark $\Delta$. The procedure is over when no more states get marked. Note that marking is carried out with respect to a fixed eventuality
ξ and is, therefore, repeated each time we want to check realization of an eventuality (see reasons further).

We have so far described elimination rules; to describe the state elimination phase as a whole, we need to specify the order of their application. We have to be careful since, having applied (E2), we could have removed all the states accessible from some Δ along the arrows marked with some formula χ; hence, we need to reapply (E1) to the resultant tableau to remove such Δ’s. Conversely, after having applied (E1), we could have thrown away some states that were needed for realizing certain eventualities; hence, we need to reapply (E2). Moreover, we cannot terminate the procedure unless we have checked that all eventualities are realized. Therefore, we apply (E1) and (E2) in a dovetailed sequence that cycles through all the eventualities. More precisely, we arrange all eventualities occurring in the states of $T^0_\theta$ in a list $\xi_1, \ldots, \xi_m$. Then, we proceed in cycles. Each cycle consists of alternatingly applying (E2) to the pending eventuality (starting with $\xi_1$), and then applying (E1) to the resulting tableau, until all the eventualities have been dealt with. These cycles are repeated until no state is removed throughout a whole cycle. When that happens, the state elimination phase is over.

The graph produced at the end of the state elimination phase is called the final tableau for $\theta$, denoted by $T^\theta$, and its set of states is denoted by $S^\theta$.

**Definition 4.9**
The final tableau $T^\theta$ is open if $\theta \in \Delta$ for some $\Delta \in S^\theta$; otherwise, $T^\theta$ is closed.

The tableau procedure returns ‘no’ (not satisfiable) if the final tableau is closed; otherwise, it returns ‘yes’ (satisfiable) and, moreover, provides sufficient information for producing a finite model satisfying $\theta$; that construction is sketched in Section 5.2.

**Example 4.10**
We will continue to make the final tableau for the formulae $\Theta_1$ considered in Example 4.6 and Example 4.7. The state elimination procedure starts with the initial tableau given in Figure 4. During the state-elimination phase, state $\Delta_1$ gets removed due to (E1), since it does not have any successor states along an arrow labelled with χ, while states $\Delta_2, \Delta_3, \Delta_4$ and $\Delta_5$ are eliminated due to (E2), as all of them contain the unrealized eventuality $\neg C_{\{a, b\}}p$. Thus, the final tableau for $\Theta$ is an empty graph; therefore, $\Theta$ is unsatisfiable.

5 Soundness and completeness of the tableau

5.1 Soundness

Technically, soundness of a tableau procedure amounts to claiming that if the input formula $\theta$ is satisfiable, then the final tableau $T^\theta$ is open.

Before going into the technical details, we give an informal outline of the proof. The tableau procedure for the input formula $\theta$ starts off with creating a single prestate $\{\theta\}$. Then, we expand $\{\theta\}$ into states, each of which contains $\theta$. To establish soundness, it suffices to show that at least one of these states survives to the end of the procedure and is, thus, part of the final tableau.

We start out by showing (Lemma 5.1) that if a prestate $\Gamma$ is satisfiable, then at least one state created from $\Gamma$ using (SR) is also satisfiable. In particular, this ensures that if $\theta$ is satisfiable, then so is at least one state obtained by (SR) from $\{\theta\}$. To ensure soundness, it suffices to prove that this state never gets eliminated from the tableau.
To that end, we first show (Lemma 5.2) that, given a satisfiable state $\Delta$, all the prestates created from $\Delta$ in accordance with (DR)—each prestate being associated with a formula of the form $\neg D_\phi$—are satisfiable; according to Lemma 5.1 each of these prestates will give rise to at least one satisfiable state. It follows that, if a tableau state $\Delta$ is satisfiable, then every successor of $\Delta$ in the initial tableau will have at least one satisfiable successor reachable by an arrow associated with each formula of the form $\neg D_\phi$ belonging to $\Delta$. Hence, if $\Delta$ is satisfiable, it will not be eliminated on account of (E1).

Second, we show that no satisfiable states contain unrealized eventualities (in the sense of Definition 4.8), and thus cannot be removed from the tableau on account of (E2). Thus, we show that a satisfiable state of the pretableau (equivalently, initial tableau) cannot be removed on account of any of the state elimination rules and, therefore, survives to the end of the procedure. In particular, this means that at least one state obtained from the initial prestate $\theta$, and thus containing $\theta$, survives to the end of the procedure. Hence, the final tableau for $\theta$ is open, as desired.

We emphasize again that the claims mentioned above, and their proofs, do not depend on the application (or not) of the cut rules $C_{\Sigma 1}$ and $C_{\Sigma 2}$ because they are sound, since $\gamma \lor \neg \gamma$ is valid for any formula $\gamma$. Therefore, these results are unaffected by the restrictive conditions $C_1$ and $C_2$ for their application.

We now proceed with the technical details.

**Lemma 5.1**
Let $\Gamma$ be a prestate of $\mathcal{P}^0$ such that $\mathcal{M}, s \models \Gamma$ for some CMAEM $\mathcal{M}$ and $s \in \mathcal{M}$. Then:

1. $\mathcal{M}, s \models \Delta$ holds for at least one $\Delta \in \text{states}(\Gamma)$.
2. Moreover, if $\neg C_\phi \in \Gamma$ and $\mathcal{M}, s \models \phi$, then $\Delta$ can be chosen so that $\neg \phi \in \Delta$.
3. If $\neg C_\phi \in \Gamma$ while none of $\neg C_\phi$’s $\beta$-components are in $\Gamma$, then for every $a \in A$, if $\mathcal{M}, s \models \neg D_a C_\phi$ then $\Delta$ can be chosen so that either $\neg D_a C_\phi \in \Delta$ or $\neg \phi \in \Delta$.

**Proof.** Straightforward from the definition of $CSE(\Gamma)$ and using Proposition 4.8. 

**Lemma 5.2**
Let $\Delta \in S_0^\mathcal{M}$ be such that $\mathcal{M}, s \models \Delta$ for some CMAEM $\mathcal{M}$ and $s \in \mathcal{M}$, and let $\neg D_\phi \in \Delta$. Then, there exists $t \in \mathcal{M}$ such that $(s, t) \in R^D_\mathcal{M}$ and $\mathcal{M}, t \models \Gamma$, for a set $\Gamma$ defined according to the rule (DR) applied to $\Delta$ and $\neg D_\phi$:

$$\Gamma = \{\neg \phi\} \cup \{D_\phi' \psi \in \Delta \mid A' \subseteq A\} \cup \{\neg D_\phi' \psi \in \Delta \mid A' \subseteq A\} \cup \{\neg D_\phi' \psi \not\in D_\phi' \psi \} \cup \{\neg D_\phi' \psi \in \Delta \mid A' \cap A \neq \emptyset\}$$

**Proof.** Easily follows from the semantics of the epistemic operators and the definition of CMAEM. 

**Lemma 5.3**
Let $\Delta \in S_0^\mathcal{M}$, let $\neg C_\phi, \neg D_a C_\phi \in \Delta$, and let, furthermore, $\Delta \xrightarrow{-D_a C_\phi} \Gamma$ for some prestate $\Gamma \in \mathcal{P}^0$. Assume that $\mathcal{M}, s \models \Delta$ and $(s, s') \in R^D_\mathcal{M}$, for some model $\mathcal{M}$ and a pair of states $s, s' \in \mathcal{M}$; then $\mathcal{M}, s' \models \Gamma$.

**Proof.** Recall from the rule (DR) that $\Gamma = \{\neg C_\phi\} \cup \{D_\phi' \mid D_\phi' \in \Delta\} \cup \{\neg D_\phi' \not\in D_\phi' \psi \} \cup \{\neg C_\phi' \mid C_\phi' \in \Delta\}$. The claim follows easily, because $R^D_\mathcal{M}$ is an equivalence relation. Indeed, $\mathcal{M}, s' \models \neg C_\phi$ because every $A$-reachable state from $s$ is $A$-reachable from $s'$, too. Moreover, $(s', s'') \in R^D_\mathcal{M}$ iff $(s, s'') \in R^D_\mathcal{M}$, for all $s''$. Therefore, $\mathcal{M}, s' \models \chi$ for all $\chi \in \Delta \setminus \{\neg C_\phi\}$. 

**Lemma 5.4**
Let $\Delta \in S_0^\mathcal{M}$ be such that $\mathcal{M}, s \models \Delta$ for some CMAEM $\mathcal{M}$ and $s \in \mathcal{M}$, and let $\neg C_\phi \in \Delta$. Then there is a finite path in $\mathcal{S}_0^\mathcal{M}$ of satisfiable states that realizes $\neg C_\phi$ at $\Delta$. 

...
The claim then follows from Lemma 5.1.

**Theorem 5.5**

**Proof.** We start by proving the following:

Let \( \neg C_\alpha \psi \in \Gamma_1 \) for some prestate \( \Gamma_1 \in \mathcal{P}^0 \) such that \( \Gamma_1 \) does not contain any of the \( \beta \)-components of \( \neg C_\alpha \psi \). Suppose that \( M, s_1 \models \Gamma_1 \), and let \( s_1 \rightarrow_{a_1}^\ldots \rightarrow_{a_n} s_n \) be a shortest path in \( M \) that satisfies \( \neg C_\alpha \psi \), i.e. \( M, s_n \models \neg \psi \), and for all \( i < n \), the following hold: \( M, s_i \models \{ \neg C_\alpha \psi, \phi \} \), and \( (s_i, s_{i+1}) \in \mathcal{R}^0_{\beta} \). Then there exists a path

\[
\Delta_1 \xrightarrow{D_{a_n} C_\psi} \Delta_2 \xrightarrow{D_{a_{n-1}} C_\psi} \ldots \xrightarrow{D_{a_1} C_\psi} \Delta_{\psi'}
\]

of satisfiable states in \( S_0^\psi \), where \( n' \leq n \), \( \Delta_1 \in \text{states}(\Gamma_1) \) and \( \neg \psi \in \Delta_{\psi'} \).

We prove the above claim by induction on \( n \).

If \( n = 1 \), then \( M, s_1 \models \neg \psi \). Since \( \neg C_\alpha \psi \in \Gamma_1 \) and \( M, s_1 \models \Gamma_1 \), Lemma 5.4 implies that there is a \( \Delta_1 \in \text{states}(\Gamma_1) \) such that \( M, s_1 \models \Delta_1 \) and \( \neg \psi \in \Delta_1 \). Thus \( \Delta_1 \) is the needed path in \( S_0^\psi \) that satisfies the claim above.

Assume now the claim holds for all \( m < n \). Let \( \neg C_\alpha \psi \in \Gamma_1 \), let \( M, s_1 \models \Gamma_1 \), and assume that none of \( \neg C_\alpha \psi \)'s \( \beta \)-components are in \( \Gamma_1 \). Let the path in \( M \) satisfying the eventuality \( \neg C_\alpha \psi \) be \( s_1 \rightarrow_{a_1}^\ldots \rightarrow_{a_n} s_n \), where \( n > 1 \).

Since \( M, s_1 \models \{ \neg D_{a_n} C_\psi, \neg C_\alpha \psi \} \), Lemma 5.4 implies the existence of \( \Delta_1 \in \text{states}(\Gamma_1) \) in \( S_0^\psi \) with \( M, s_1 \models \Delta_1 \), and \( \neg D_{a_n} C_\psi \in \Delta_1 \) or \( \neg \psi \in \Delta_1 \). In the latter case, \( \Delta_1 \) is the needed path. In the former case, due to Lemma 5.3 there exists a prestate \( \Gamma_2 \in \mathcal{P}^0 \), with \( \Delta_1 \xrightarrow{D_{a_n} C_\psi} \Gamma_2 \); then, \( \neg C_\alpha \psi \in \Gamma_2 \). Note that \( \Gamma_2 \) cannot contain any of \( \neg C_\alpha \psi \)'s \( \beta \)-components, since \( \Delta_1 \) contains \( \neg D_{a_n} C_\psi \), and thus, it can contain at most one other \( \beta \)-component, namely \( \neg \psi \). But in that case we would have that \( M, s_1 \models \neg \psi \), which contradicts the assumption. Lemma 5.3 gives us \( M, s_2 \models \Gamma_2 \).

Thus, since \( s_2 \rightarrow_{a_2}^\ldots \rightarrow_{a_n} s_n \) is a path of length \( n - 1 \) that realizes \( \neg C_\alpha \psi \) at \( s_2 \), the induction hypothesis claims that there is a path of satisfiable states in \( S_0^\psi \),

\[
\Delta_2 \xrightarrow{D_{a_n} C_\psi} \ldots \xrightarrow{D_{a_{n-1}} C_\psi} \Delta_{\psi'}
\]

where \( n' \leq n - 1 \), \( \Delta_2 \in \text{states}(\Gamma_2) \), \( \neg \psi \in \Delta_{\psi'} \).

Since \( \Gamma_1 \xrightarrow{\Delta_1 \rightarrow} \Gamma_2 \xrightarrow{\Delta_2} \), we obtain a path in \( S_0^\psi \) of length at most \( n \) that satisfies the induction hypothesis.

That concludes the induction.

Getting back to the claim of the Lemma, we have that if \( \neg C_\alpha \psi \in \Delta \), then either \( \neg \psi \in \Delta \) or there exists an \( a' \in A \) such that \( \neg D_{a'} C_\psi \in \Delta \), since \( \Delta \) is fully expanded. In the former case, \( \neg C_\alpha \psi \) is realized in \( \Delta \) and the claims follows. In the latter case, there will be a prestate \( \Gamma \in \mathcal{P}^0 \), such that \( \Delta \xrightarrow{D_{a'} C_\psi} \Gamma \). Note that in this case \( \Gamma \subseteq \Delta \). Due to (DR) and the fact that \( \neg \psi \notin \Delta, \Gamma \) cannot contain any of \( \neg C_\alpha \psi \)'s \( \beta \)-components.

Thus, the statement above gives us that there is a path of satisfiable states in \( S_0^\psi \), that realizes \( \neg C_\alpha \psi \in \Delta \).

**Theorem 5.5**

If \( \theta \in \mathcal{L} \) is satisfiable in a CMAEM, then \( T^0 \) is open.

**Proof.** Using the preceding Lemma 5.3 and Lemma 5.4, one can show by induction on the number of stages in the state elimination process that no satisfiable state can be eliminated due to (E1)--(E2). The claim then follows from Lemma 5.3.
5.2 Completeness

The completeness of a tableau procedure means that if the tableau for a formula $\theta$ is open, then $\theta$ is satisfiable in a CMAEM. In view of Theorem 3.15, it suffices to show that an open tableau for $\theta$ can be turned into a CMAEHS for $\theta$. In order to prove that, we need to specify sufficiently strong restrictive conditions $C_1$ and $C_2$ governing the application of the cut rules $CS_1$ and $CS_2$ respectively on formulas $D_A\phi$ and $C_A\phi$ in the Definition 4.1 of cut-saturated sets. We now specify these conditions as follows.

$C_1$ Cut on $D_A\psi \in Sub(\psi)$ where $\psi \in \Delta$, if either of the following holds:
- $C_{11}$: $\psi = D_B \delta$ or $\psi = \neg D_B \delta$, and there is a $\neg D_E \epsilon \in \Delta$ such that $A \subseteq E$ and $B \subseteq E$.
- $C_{12}$: $\psi = \neg C_B \delta$ and there exists a $\neg D_E \epsilon \in \Delta$ such that $A \subseteq E$ and $B \cap E \neq \emptyset$.

$C_2$ Cut on $C_A\phi \in Sub(\psi)$ where $\psi \in \Delta$, if either of the following holds:
- $C_{21}$: $\psi = D_B \delta$ or $\psi = \neg D_B \delta$, and there exists a $\neg D_E \epsilon \in \Delta$ such that $B \subseteq E$ and $A \cap E \neq \emptyset$.
- $C_{22}$: $\psi = \neg C_B \delta$ and there exists a $\neg D_E \epsilon \in \Delta$ such that $B \cap E \neq \emptyset$.

The intuition: a cut rule only has to be applied to a formula $D_A\psi$ or $C_A\psi$ if:

(i) that formula can occur in the label of a descendant state and,
(ii) once it occurs there, it will have an effect spreading back to the current state.

For the former to happen, that formula must occur in a $D_B$-formula or a $\neg D_B$-formula or a $\neg C_B$-formula. For the latter, the path leading from the current state to that descendant must be labelled with relations propagating the effect of the respective box.

**Example 5.6**

This example illustrates the need for applying cut rules and using cut-saturated sets instead of simply fully expanded sets. First, recall the requirement of the relations in a (pseudo-)CMAEL(CD) model to be equivalence relations, reflected in $(CH3)$ of Definition 3.9 for Hintikka structures. Now, consider the tableau constructed for the formula $\theta = \neg D_{\{a,b\}} p \land \neg D_{\{a,c\}} \neg D_a p$ if we would only use fully expanded sets:

The corresponding claimed Hintikka structure and (pseudo)-model, that this tableau would produce (see the construction in Lemma 5.8) would then be, respectively:

In the ‘Hintikka’-structure to the left, we have that $D_a p$ is in the state in the bottom right corner, but not in the state in the top, though the edge connecting them is labelled with $\{a,c\}$. This on the
other hand means, that \( \theta \) is not satisfied in the ‘model’ to the right, because \( \neg D_{ap} \) does not hold at any state, hence \( \neg D_{ap} \) in not true at any state. In fact, \( \theta \) is not satisfiable at all.

If we would indeed apply the cut-rules then the tableau for \( \theta \) would close. The pretableau for \( \theta \) would look as follows.

Notice that some of the prestates (namely \( \neg p, \neg D_{ap} \)) do not have any full expansions since these are patently inconsistent. After the initial tableau has been build, this then causes the two states in \( \text{states}(\theta) \) to be deleted by \( \text{(E1)} \) and the final tableau is

\[
\neg p, \neg D_{ap} \rightarrow \neg p, D_{ap}, p \rightarrow \neg p
\]

which closes.

The following lemma is needed to ensure that the satisfaction of the condition \( [CH3] \) from the definition of Hintikka structures for \( \text{CMAEL(CD)} \) is guaranteed in the final tableau.

**Lemma 5.7**

Suppose \( \Delta \models D_{ap} \psi \rightarrow \Delta' \) in the final tableau \( T^\theta \) for some input formula \( \theta \) and suppose that \( D_{ap} \psi \in \Delta' \) where \( B \subseteq A \). Then \( D_{ap} \psi \in \Delta \).

**Proof.** First, note that if the cut rules \( [CS1] \) and \( [CS2] \) are applied unrestrictedly to every subformula \( D_{ap} \psi \) or \( C_{ap} \psi \) of a formula in the label of the current state \( \Delta \), the proof of the lemma is immediate. We will show that the claim still holds if the restrictions \( C_1 \) and \( C_2 \), specified above, are imposed.

For a formula \( \alpha \) we let \( CS_1(\alpha) \) be the set of all formulae that can occur in any one-step cut-saturated expansion of \( \alpha \) according to the procedure described in Definition 4.2. Similarly \( CS_1(\Delta) = \bigcup_{\alpha \in \Delta} CS_1(\alpha) \) for a set \( \Delta \) of formulae, and recursively we let \( CS_n(\Delta) = CS_1(CS_{n-1}(\Delta)) \). As is easy to see, this construction converges, and the following is true:

- For any formula \( \alpha \) and any \( n \in \mathbb{N}, \) \( CS_n(\alpha) \subseteq \text{ec}(\alpha), \) i.e.:
  \[
  CS_n(\alpha) \subseteq \{ \beta, \neg \beta \mid \beta \in \text{Sub}(\alpha) \} \cup \{ D_{ap} C_{ap}, \neg D_{ap} C_{ap} \mid C_{ap} \in \text{Sub}(\alpha), e \in E \}
  \]

- For any cut-saturated expansion \( \Omega \) of \( \Gamma \) there is an \( n \) such that \( \Omega \subseteq CS_n(\Gamma) \).

If \( \beta \in CS_n(\Gamma) \), then there is an \( \alpha \in \Gamma \), such that \( \beta \in CS_n(\alpha) \).

Now, let \( \Gamma \) be the prestate in the pretableau \( T^\theta \) that gives rise to the relation between \( \Delta \) and \( \Delta' \), i.e. \( \Delta \models D_{ap} \psi \rightarrow \Delta' \) in \( T^\theta \). The above gives us that since \( \Delta' \) is a cut-saturated expansion of \( \Gamma \) and \( D_{ap} \psi \in \Delta' \), there is an \( \alpha \in \Gamma \) such that \( D_{ap} \psi \in \text{ec}(\alpha) \). That is, either \( D_{ap} \psi \in \text{Sub}(\alpha) \), or \( D_{ap} \psi = D_{ap} C_{ap} \delta \) for a \( C_{ap} \delta \in \text{Sub}(\alpha) \) and a \( d \in D \).
Since $\alpha \in \Gamma$, due to (DR), either $\alpha = D_C \gamma \in \Delta$ or $\alpha = -D_C \gamma \in \Delta$ for a $C \subseteq A$, or $\alpha = -C_C \gamma \in \Delta$ where $C \cap A \neq \emptyset$, or $\alpha = \neg \gamma$. We notice that it is enough to show that $[\alpha]$ is applicable to $D_\theta \psi$ at $\Delta$, since then either $D_\theta \psi \in \Delta$ (which is what we want) or $\neg D_\theta \psi \in \Delta$; the latter would, according to (DR), imply that $\neg D_\theta \psi \in \Gamma \subseteq \Delta'$, which would cause $\Delta'$ to be patently inconsistent, which contradicts $\Delta'$ being a cut-saturated set and thus fully expanded (cf. CS0). We split according to cases:

**Case 1.** $\alpha = D_C \gamma \in \Delta$ or $\alpha = -D_C \gamma \in \Delta$ for a $C \subseteq A$: $D_\theta \psi \in \Theta C_\theta (D_C \gamma)$ gives that $D_\theta \psi \in \Theta C_\theta (D_C \gamma)$, or $D_\theta \psi = D_\gamma C_\delta$ for a $C_\delta \in \Theta (D_C \gamma)$, where $d \in D$.

In the first case, $D_\theta \psi$ is a subformula of a $D_C$-formula in $\Delta$, and since $C, B \subseteq A$ and $\neg D_\delta \psi \in \Delta$, $[\alpha]$ is applicable to $D_\theta \psi$ at $\Delta$.

In the second case, $D_\theta \psi = D_\gamma C_\delta$ for an $C_\delta \in \Theta (D_C \gamma)$ with $d \in D$. Since $B = \{d\} \subseteq A$, we have $d \in D \cap A$ and hence $[\alpha]$ is applicable to $C_\delta \in \Delta$, as we also have $C \subseteq A$ and $\neg D_\delta \psi \in \Delta$. This means that either $C_\delta \in \Delta$ or $\neg C_\delta \in \Delta$ according to CS0. If $C_\delta \in \Delta$, then $D_\theta \psi = D_\delta C_\delta \in \Delta$ according to CS0. If $\neg C_\delta \in \Delta$, then according to (DR), $\neg C_\delta \in \Gamma$ since $d \in D \cap A$. However, $D_\theta \psi = D_\delta C_\delta \in \Gamma$, and hence $C_\delta \in \Delta$. This gives us a contradiction, as $\Delta'$ is fully expanded and, thus, not patently inconsistent.

The case where $\alpha = -D_C \gamma$ is similar.

**Case 2.** $\alpha = \neg \gamma$: $D_\theta \psi \in \Theta C_\theta (\neg \gamma)$. We have two cases to consider:

Either $D_\theta \psi \in \Theta C_\theta (\neg \gamma)$, in which case $D_\theta \psi \in \Theta C_\theta (\neg \gamma)$ and thus $[\alpha]$ is applicable to $D_\theta \psi$ at $\Delta$.

As before, the former implies that $D_\theta \psi \in \Delta$, as desired, while the latter leads to a contradiction.

**Lemma 5.8**

If $T^{\theta}$ is open, then there exists a CMAEHS for $\theta$.

**Proof.** The needed Hintikka structure $H$ for the formula $\theta$ is built out of the final tableau $T^{\theta}$ by renaming the relations between the states, such that they correspond to a subset of $\Sigma$. This is done by labelling the edges from $\Delta$ to $\Delta'$ with the set $A$ for which $\Delta \xrightarrow{\neg D_\psi \theta} \Delta'$ in $T^{\theta}$.

Now, let $\Sigma$ be the set of agents occurring in $\theta$, and let $S = S^{\theta}$. For any $A \in \mathcal{P}^+(\Sigma)$, let $R_A^{D} = \{(\Delta, \Delta') \in S \times S | \Delta \xrightarrow{\neg D_\psi \theta} \Delta' \}$ for some $\psi$, and let $R_A^{\Delta}$ be the reflexive, transitive closure of $\cup_{B \subseteq A} R_B^{D}$. Let $L(\Delta)$ be the labelling of the state in $T$, i.e., the sets of formulae that has been associated with $\Delta$.

Finally, let $H_\theta = (\Sigma, S^{\theta}, \{R_A^{D} \}_{A \in \mathcal{P}^+(\Sigma)}, \{R_A^{\Delta} \}_{A \in \mathcal{P}^+(\Sigma)}, A \mathcal{P}, L)$.

We will now show that $H_\theta$ is a Hintikka structure. To that end, we have to prove $(\Sigma, S, \{R_A^{D} \}_{A \in \mathcal{P}^+(\Sigma)}, \{R_A^{\Delta} \}_{A \in \mathcal{P}^+(\Sigma)})$ is a CMAES, and that conditions [(CH1)][(CH4)] of Definition 3.3 hold for $H$. The former is clear from the construction of $H$.

**[CH1]** holds since all states in the final tableau are fully expanded.

**[CH2]** is satisfied since, otherwise, the state would have been deleted from the tableau due to (E1).

Likewise, **[CH4]** is satisfied since, otherwise, the state would have been removed due to (E2).

It remains to show that **[CH3]** holds. Let $(\Delta, \Delta') \in R_D$ (i.e., $\Delta \xrightarrow{\neg D_\psi \theta} \Delta'$ in $T^{\theta}$), and $B \subseteq A$. We need to show that $D_\theta \psi \in \Delta \Rightarrow D_\theta \psi \in \Delta'$. If $D_\theta \psi \in \Delta$, then due to the propagation rule (DR), $D_\theta \psi \in \Gamma$, and...
where $\Gamma$ is the prestate in the final pretableau, such that $\Delta \xrightarrow{D_A \psi} \Gamma \rightarrow \Delta'$. Thus $D_B \psi$ is also in $\Delta'$ since $\Gamma$ is included in all cut-saturated expansions of $\Gamma$. The other direction follows from Lemma 5.8.

**THEOREM 5.9 (Completeness)**

Let $\theta \in \mathcal{L}$ and let $\mathcal{T}^\theta$ be open. Then, $\theta$ is satisfiable in a CMAEM.

**PROOF.** Immediate from Lemma 5.8 and Theorem 3.15.

### 6 Complexity, efficiency and possible optimizations of the tableau procedure

#### 6.1 Complexity

The termination of the tableau procedure described above is a fairly straightforward consequence of the finiteness of the set of all possible labels of states and prestates and their re-use in the construction phase. In this subsection, we estimate the worst-case running time of all phases of the procedure.

We denote by $|\theta|$ the length of a formula $\theta$ and by $|\mathcal{Ecl}(\theta)|$ the number of formulae in $\mathcal{Ecl}(\theta)$. Let $|\theta| = n$ and the number of agents occurring in $\theta$ be $k$.

By Lemma 4.5, $|\mathcal{Ecl}(\theta)| \leq ckn$ for some (small) constant $c$. Then, the number of prestates and states in the tableau for $\theta$ is $O(2^{ckn})$. Comparing two states or prestates takes $O(kn)$ steps (assuming a fixed order of the formulae in $\mathcal{Ecl}(\theta)$, and each state being represented as a $0/1$ string of length $ckn$), hence checking whether a prestate or a state has already been created, takes $O(kn2^{ckn})$. Therefore, the construction phase takes time $O(kn2^{ckn})$.

The prestate elimination phase takes time $O(2^{ckn})$. Checking realization of an eventuality in a state takes $O(kn2^{ckn})$ steps and the number of eventualities is bounded by $n$, hence the elimination of a ‘bad’ state takes at most $O(n2^{ckn})$ steps. Hence, the elimination state takes $O(n2^{ckn})$ steps.

We conclude that the whole tableau procedure terminates in $O(n2^{ckn})$ steps, hence it is in EXPTIME, which is in compliance with the known EXPTIME(-complete) lower bound (see [9], [10]).

#### 6.2 Efficiency

Some features of the ‘diamond-propagating’ procedure described above make it sometimes practically sub-efficient.

Firstly, the application of the cut rules $\text{CS1}$ and $\text{CS2}$ can produce many cut-saturated sets, even after imposing the restrictive conditions $C_1$ and $C_2$. Potentially, it can create a number of states that is exponential in the number of subformulae of the form $D_A \psi$ or $C_A \psi$ occurring in the formulas of the input set $\Gamma$.

Secondly, when applying the rule $\text{(DR)}$ to a state $\Delta$ with respect to some $\neg D_A \psi$, we propagate to the newly created prestate all the diamond-formulae of the form $\neg D_B \psi$, where $B \subseteq A$, except $\neg D_A \psi$ itself. Likewise, all formulae $\neg C_A \psi$ where $A$ and $B$ are not disjoint, get propagated. Thus, the presence of a ‘diamond’ in a prestate $\Gamma$ is then passed on to all states in $\text{states}(\Gamma)$, resulting in the need to apply the rule $\text{(DR)}$ to every state in $\text{states}(\Gamma)$ with respect to this diamond; this, again, implies the creation of a large number of states (even though, as we have shown, the maximal number of states is still no more than exponential in the size of the input formula). However, we re-iterate that this ‘diamond-propagation’ is necessary for the procedure developed here, because if a diamond-formula is not propagated forward, then its negation, which is a box-formula, may be added to a successor state and thus clash with that diamond-formula in the current state.
Tableaux for the full coalitional multiagent epistemic logic

On the other hand, the restrictive conditions $C_1$ and $C_2$ for the application of cut-saturation in the production of CS-expansions can have a very significant effect on the size of the tableau, as illustrated by the next example.

**Example 6.1**

Suppose we want to build a tableau for the formula $\theta = C_{(a,b)}D_x p \rightarrow \neg C_{(b,c)}D_y p \equiv \neg (C_{(a,b)}D_x p \land C_{(b,c)}D_y p)$ and suppose that $\Sigma = \{a, b, c\}$. We start off with creating a single prestate $[\theta]$. Using only the unrestricted conditions $C_1$ and $C_2$ to cut, applying the rule (SR) to this prestate produces an overwhelming number of 35 states:

1. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p]$;
2. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
3. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
4. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
5. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
6. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
7. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
8. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
9. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
10. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
11. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
12. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
13. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
14. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
15. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
16. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
17. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
18. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
19. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
20. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
21. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
22. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
23. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
24. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
25. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
26. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
27. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
28. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
29. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
30. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
31. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
32. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
33. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
34. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;
35. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, \neg C_{(b,c)}D_y p, \neg D_y p]$;

If we instead use the restricted $C_1$ and $C_2$ we will produce 8 states:

1. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, p]$;
2. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, p]$;
3. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, p]$;
4. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, p]$;
5. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, p]$;
6. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, p]$;
7. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, p]$;
8. $[\theta, \neg C_{(a,b)}D_x p, \neg D_x C_{(a,b)}D_y p, D_x p, p]$. 
Tableaux for the full coalitional multiagent epistemic logic

Figure 5 shows the pretableau for one part of $\theta$, i.e. $\neg C_{\{a,b\}}Dap$. The tableau for the other part of $\theta$ will be similar and disjoint from this tableau.

As seen here, the backtracking procedure is rather inefficient when applied to formulae of the type of $C_{\{a,b\}}Dap \rightarrow \neg C_{\{b,c\}}Dbp$.

Both causes of potential inefficiencies discussed above, viz. the forward diamond-propagation and the (restricted) analytic cut rules on box-formulae, are needed to ensure that every ‘successful’ tableau can be turned into a Hintikka structure. More precisely, they together ensure that the right-to-left implication in the statement of property (CH3) of Hintikka structures (recall Definition 3.9) holds.

A possible way of eliminating these causes for inefficiencies is to change the strategy in the tableau-building, by implementing a mechanism for backward propagation of boxes: if $Dap$ occurs in a state $\Delta$, then ensure that this box is propagated backwards to all predecessor states where it must occur. The main disadvantage of this approach is that it requires an elaborated mechanism of repeated updating the hitherto constructed part of the tableau. We leave the realization of this idea for future work.

6.3 Improvements

As stated earlier, the main emphasize of our tableau construction is the ease of presentation, comprehension and implementation, rather than technical sophistication and optimality of the procedure. While being worst-case time optimal, it is amenable to various improvements and further optimizations, some of which we will mention briefly here.
To begin with, for methodological reasons, our procedure is divided into three phases, where the different components of the tableau-building procedure are dealt with separately. That separation of the procedure into phases makes it less optimal compared to the approach whereby the three phases are carried out simultaneously and the prestate and state elimination is done ‘on-the-fly’.

Also, as briefly mentioned in Section 4.1 it is possible to make the procedure cut-free by using a mechanism for ‘backwards propagation’ of $D$-formulas, which, when well designed can lead to more optimal performance in some cases. This approach is taken e.g. in [22], where the authors construct a cut-free tableau-based algorithm for the logic PDL with converse, while the algorithm presented in [29] builds on this work by constructing a cut-free tableaux-based algorithm for the description logic SHI, which contains inverse roles. Both methods account for the case where a (number of) formula(s) turns up in a node, which will be required to be in the already created predecessor node of the node in question. The former algorithm deals with eventualities, too. Adopting this approach to our procedure while optimizing it for the logic CMAEL(CD) would result in a procedure sketched below.

6.3.1 State elimination ‘on-the-fly’

Here we make use of the concept of ‘potential rescuers’ used in [22] and [29], though in a slightly different way, adjusted to our needs. We likewise take on board the techniques of updating and propagating statuses of nodes in the tableaux.

Firstly, we maintain a status for (pre)states, which can either be unexplored, open or closed. The status of a (pre)state is initially set to unexplored when the (pre)state is created, and then updated during the procedure. When a prestate is expanded or a state expanded for all diamond-formulas in it, its status changes to open. Later on the status of a state $\Delta$ can then change to closed in the following cases:

- there is an epistemic prestate $\Gamma$ such that $\Delta \rightarrow \delta \rightarrow \Gamma$ for a formula $\delta$ and the status of $\Gamma$ is closed.
- $\Delta$ contains an eventuality $\neg C A \phi$ that it neither realized in the current tableau under construction nor has a ‘potential rescuer’. A potential rescuer is a (pre)state, which is $A$-reachable from $\Delta$, contains $\neg C A \phi$, and has not been expanded yet, i.e. it has status unexplored. Here we use a modified definition of $A$-reachability, where $\leftrightarrow$-arrows are allowed too.

The status of a prestate $\Gamma$ is set to closed if:

- all states in $\text{states}(\Gamma)$ are closed, including the case where $\text{states}(\Gamma) = \emptyset$, or
- $\Gamma$ contains an eventuality that it neither realized nor has a potential rescuer.

Additionally, we make sure, that unsatisfiable (pre)states are removed on-the-fly and that the procedure stops and the tableau closes as soon as unsatisfiability of the input prestate is detected during the procedure, i.e.:

- We close a prestate when it is expanded and does not have any cut-saturated expansions.
- When a (pre)state $\Sigma$ closes, we propagate updates of statuses to the relevant (pre)states, whose status depend on the status of $\Sigma$. These are (pre)states that have outgoing arrows pointing to $\Sigma$.
- We keep an eye on the initial prestate, labelled with the input formula whose satisfiability we are checking. When/if this prestate closes, we stop the whole procedure and return ‘unsat’.
Finally, we also want to avoid the unnecessary checking of unrealized eventualities, since this step is one of the more expensive checks. Thus, when updating the status of a (pre)state we only check containment of unrealized eventualities, when this is really necessary. E.g. we do not check that if a potential rescuer is known to be reachable. This of course requires some bookkeeping.

6.3.2 Making the procedure cut-free

The procedure above takes care of doing the satisfiability checking ‘on the fly’, however it is not cut-free. Though, the procedure can be made cut-free by incorporating the following:

Firstly, we use full expansions instead of cut-saturated expansions. Secondly, we now need to account for a further reason why a state $\Delta$ can close, namely that $\Delta$ contains a diamond formula $\neg D^{\psi}$, such that $\Delta \rightarrow^{D^{\psi}} \Gamma$ and all states in $\text{states}(\Gamma)$ are incompatible with $\Delta$ with respect to $\neg D^{\psi}$. Here, $\Delta' \in \text{states}(\Gamma)$ is incompatible with $\Delta$ if $\{ D^{\psi} \in \Delta' | A' \subseteq A \} \subseteq \Delta$, i.e. condition $[\text{CH3}]$ will not be fulfilled in the resulting Hintikka structure. This, however, does not necessarily mean, that the state $\Delta$ needs to close. After all, since we are not proactively looking ahead for box-formulas which could possibly occur in a future descendent state of $\Delta$ and include these in $\Delta$ (as is done when using cut-saturated expansions), it is possible that $\Delta$ could become satisfiable if the box-formulas in question were added to $\Delta$.

Therefore, when it happens that $\Delta \rightarrow^{D^{\psi}} \Gamma$ and none of the states in $\text{states}(\Gamma)$ are compatible with $\Delta$ with respect to $\neg D^{\psi}$, we construct so-called ‘alternatives’ for the state $\Delta$. These are states labelled with the fully expanded sets $\Delta \cup S'$ for each $S' \in \bigcup_{\Delta \in \text{states}(\Gamma)} \mathcal{F}(\{ D^{\psi} \in \Delta' | A' \subseteq A \} \cup D^{\psi} \notin \Delta)$. Then $\rightarrow^{n}$-arrows pointing to these alternatives are added from each prestate pointing to $\Delta$, and finally we close the original state $\Delta$ (and propagate the change of status that hereby occurs, as described previously).

In this procedure, we need to keep track of when such incompatibilities occur, which requires some further bookkeeping.

7 Concluding remarks

We have developed a sound and complete tableau-based decision procedure for the full coalitional multiagent epistemic logic $\text{CMAEL}(\text{CD})$. The incremental tableau style adopted here is intuitive, practically more efficient, and more flexible than the maximal tableau style, developed e.g. for the fragment $\text{MAEL}(\text{C})$ of $\text{CMAEL}(\text{CD})$ in [25], and therefore it is more suitable both for manual and automated execution. In fact, an earlier, less optimal version, of this procedure has been implemented and reported in [42]. On the other hand, as discussed in the previous section, various further optimizations of the procedure are possible and desirable, and some such optimizations have been developed for logics related to $\text{CMAEL}(\text{CD})$, see Section 1.2. Furthermore, our tableau procedure is also amenable to various extensions, subject to current and future work:

- to temporal epistemic logics of linear and branching time, preliminary reports on which have appeared respectively in [15] and [16];
- with the strategic abilities operators of the Alternating-time temporal logic $\text{ATL}$, a tableau-based decision procedure for which were developed in [13]. Merging tableaux for these two logical systems will produce, inter alia, a feasible decision procedure for the Alternating-time temporal epistemic logic $\text{ATEL}$ [40];
- a cut-free, ‘on the fly’ version, as described in Section 6.3.
Acknowledgments

We gratefully acknowledge the financial support from: the FIRST Research School and Roskilde University, funding the PhD study of the first author, part of which is her contribution to the present research; the National Research Foundation of South Africa for the second author through several research grants, and the Claude Harris Leon Foundation, funding the third author’s post-doctoral fellowship at the University of the Witwatersrand, during which the initial work of this research was done. We also thank the anonymous referees for their valuable comments and constructive criticism.

References

Tableaux for the full coalitional multiagent epistemic logic


Tableaux for the full coalitional multiagent epistemic logic


Received 18 February 2011