Tableau-based decision procedures for logics of strategic ability in multiagent systems

VALENTIN GORANKO
University of the Witwatersrand
DMITRY SHKATOV
University of the Witwatersrand

We develop an incremental tableau-based decision procedures for the Alternating-time temporal logic **ATL** and some of its variants. While running within the theoretically established complexity upper bound, we believe that our tableaux are practically more efficient in the average case than other decision procedures for **ATL** known so far. Besides, the ease of its adaptation to variants of **ATL** demonstrates the flexibility of the proposed procedure.

Categories and Subject Descriptors: F.4.1 [**Mathematical logic**]: Modal logic, Temporal logic; I.2.11 [**Distributed aftificial intelligence**]: Multiagent systems

General Terms: Theory

Additional Key Words and Phrases: Logics for multiagent systems, Alternating-time temporal logic, decision procedures, tableaux

1. INTRODUCTION

Multiagent systems ([Fagin et al. 1995], [Weiss 1999], [Wooldridge 2002], [Shoham and Leyton-Brown 2008]) are an increasingly important and active area of interdisciplinary research on the border of computer science, artificial intelligence, and game theory, as they model a wide variety of phenomena in these fields, including open and interactive systems, distributed computations, security protocols, knowledge and information exchange, coalitional abilities in games, etc. Not surprisingly, a number of logical formalisms have been proposed for specification, verification, and reasoning about multiagent systems. These formalisms, broadly speaking, fall into two categories: those for reasoning about *knowledge of agents* and those for reasoning about *abilities of agents*. In the present paper, we deal with the latter variety of logics, the most influential among them being the so-called Alternating-time temporal logic (**ATL**), introduced in [Alur et al. 1997] and further developed in [Alur et al. 1998] and [Alur et al. 2002].

ATL and its modifications can be applied to multiagent systems in a similar way

Author's address: V. Goranko, School of Mathematics, University of the Witwatersrand, Private Bag X3, WITS 2050, Johannesburg, South Africa.
Author's address: D. Shkatov, School of Computer Science, University of the Witwatersrand, Private Bag X3, WITS 2050, Johannesburg, South Africa.

as temporal logics, such as **LTL** and **CTL**, are applied to reactive systems. First, since **ATL**-models can be viewed as abstractions of multiagent systems, **ATL** can be used to verify and specify properties of such systems. Given a model $\mathcal{M}$ and an **ATL**-formula φ, the task of verifying $\mathcal{M}$ with respect to the property expressed by φ is, in logical terms, the model checking problem for **ATL**, extensively discussed in [Alur et al. 2002]; a model-checker for **ATL** has also been developed, see [Alur et al. 1998]. Second, **ATL** can be used to design multiagent systems conforming to a given specification; then, **ATL**-formulae are viewed as specifications to be realized rather than verified. In logical terms, this is the *constructive satisfiability problem* for **ATL**: given a formula φ, check if it is satisfiable and, if so, construct a model of φ.

In the temporal logic tradition, in which **ATL** is rooted, two approaches to constructive satisfiability are predominant: *tableau-based* and *automata-based.* The relationship between the two is not, in our view, sufficiently well understood despite being widely acknowledged. The automata-based approach to **ATL**-satisfiability was developed in [van Drimmelen 2003] and [Goranko and van Drimmelen 2006].

The aim of the present paper is to develop practically useful "incremental" (also called "goal-driven") tableau-based decision procedures (in the style of [Wolper 1985]) for the constructive satisfiability problem for the "standard" **ATL** and some of its modifications. Incremental tableaux are one of the two most popular types of tableau-based decision procedures for modal and temporal logics with fixpoint operators (the most widely known examples of such logics being **LTL** and **CTL**). It should be noted that, while tableaux for logics with fixpoint-defined operators exhibit all common features of the "traditional" tableaux for modal logics, comprehensively covered in [Fitting 1983], [Goré 1988], and [Fitting 2007], they differ substantially from the latter, as they involve a loop-detecting (or equivalent) procedure that checks for the satisfaction of formulas containing fixpoint operators.

The alternative to the incremental tableaux for logics with fixpoint operators are the "top-down" tableaux, developed, for the case of **CTL** and some closely related logics, in [Emerson and Halpern 1985] (see also [Emerson 1990]) and essentially applied to **ATL** in [Walther et al. 2006]. A major practical drawback of the top-down tableaux is that, while they run within the same worst-case complexity bound as the corresponding incremental tableaux, their performance matches the worst-case upper bound for *every* formula to be tested for satisfiability. The reason for this "practical inefficiency" of the top-down tableaux is that they invariably involve the construction of all maximally consistent subsets of the so-called "extended closure" of the formula to be tested, which in itself requires the number of steps of the order of the theoretical upper bound[1]. Some authors consider it to be so great a disadvantage of the top-down tableaux that they propose non-optimal complexity tableaux for such logics, which they claim to perform better in practice (see [Abate et al. 2007]).

[1]It should be stressed that the top-down tableaux for **ATL** presented in [Walther et al. 2006] were not meant to serve as a practically efficient method of checking **ATL**-satisfiability, but rather were used as a tool for establishing the **ExpTime** upper bound for **ATL**, in particular, for the case when the number of agents is not fixed, as assumed in [van Drimmelen 2003] and [Goranko and van Drimmelen 2006], but taken as a parameter.

We believe that the incremental tableaux developed in the present paper are intuitively more appealing, practically more efficient, and therefore more suitable both for manual and computerized execution than the top-down tableaux, not least because checking satisfiability of a formula using incremental tableaux takes, on average, much less time than predicted by the worst-case complexity upper-bound. Furthermore, incremental tableaux are quite flexible and amenable to modifications and extensions covering not only variants of **ATL** considered in this paper, but also a number of other logics for multiagent systems, such as multiagent epistemic logics (see [Fagin et al. 1995]), for which analogous tableau-based decision procedures have recently been developed in [Goranko and Shkatov] and [Goranko and Shkatov 2008]. Lastly, it should be noted, that our tableau method naturally reduces (in the one-agent case) to incremental tableaux for **CTL**, which is practically more efficient (again, on average) than Emerson and Halpern's top-down tableaux from [Emerson and Halpern 1985].

We should also mention that yet another type of tableau-style decision procedure for **ATL**, the so-called "tableau games", has been considered in [Hansen 2004]. Even though neither soundness nor completeness of the tableau games for the full **ATL** has been established in [Hansen 2004], sound and complete tableau games for the "Next-time fragment of **ATL**", namely, the Coalition Logic **CL**, introduced in [Pauly 2001a] (see also [Pauly 2001b] and [Pauly 2002]), have been presented in [Hansen 2004].

The structure of the present paper is as follows: after introducing the syntactic and semantic basics of **ATL** in section 2, we introduce, in section 3, concurrent game Hintikka structures and show that they provide semantics for **ATL** that is, satisfiability-wise, equivalent to the one based on concurrent game models described in section 2. In section 4, we develop the tableau procedure for **ATL** and analyze its complexity, while in section 5 we prove its soundness and completeness using concurrent game Hintikka structures introduced in section 3. In section 6, we briefly discuss adaptations of our tableau method for some modifications of **ATL**.

2. PRELIMINARIES: THE MULTIAGENT LOGIC **ATL**

ATL was introduced in [Alur et al. 1997], and further developed in [Alur et al. 1998] and [Alur et al. 2002], as a logical formalism to reason about open systems ([Hewitt 1990]), but it naturally applies to the more general case of multiagent systems. Technically, **ATL** is an extension of the multiagent coalition logics **CL** and **ECL** studied in [Pauly 2001a], [Pauly 2001b], and [Pauly 2002] (for a comparison of the logics, see [Goranko 2001] and [Goranko and Jamroga 2004]).

2.1 ATL syntax

ATL is a multimodal logic with **CTL**-style modalities indexed by subsets, commonly called *coalitions*, of the finite, non-empty set of names of *agents*, or players, belonging to the language. Thus, formulae of **ATL** are defined with respect to a finite, non-empty set Σ of names of agents, usually denoted by the natural numbers 1 through $|\Sigma|$ (the cardinality of Σ), and a finite or countably infinite set $\mathtt{AP}$ of atomic propositions.

Definition 2.1. **ATL**-formulae are defined by the following grammar:

$$\varphi := p \mid \neg\varphi \mid (\varphi_1 \to \varphi_2) \mid \langle\langle A\rangle\rangle\bigcirc\varphi \mid \langle\langle A\rangle\rangle\Box\varphi \mid \langle\langle A\rangle\rangle\varphi_1\mathcal{U}\varphi_2,$$

where p ranges over $\mathtt{AP}$ and A ranges over $\mathcal{P}(\Sigma)$, the power-set of Σ.

Notice that we allow (countably) infinitely many propositional parameters, but in line with traditional presentations of **ATL** (see, for example, [Alur et al. 2002]), only finitely many names of agents. We will show, however, after introducing **ATL**-semantics, that this latter restriction is not essential (see Remark 2.16 below) and thus does not result in a loss of generality.

The other boolean connectives and the propositional constant $\top$ ("truth") can be defined in the usual way. Also, $\langle\langle A\rangle\rangle\Diamond\varphi$ can be defined as $\langle\langle A\rangle\rangle\top\mathcal{U}\varphi$. As will become intuitively clear from the semantics of **ATL**, $\langle\langle A\rangle\rangle\Diamond\varphi$ and $\langle\langle A\rangle\rangle\Box\varphi$ are not interdefinable[2].

The expression $\langle\langle A\rangle\rangle$, where $A \subseteq \Sigma$, is a *coalition quantifier* (also referred to in the literature as "path quantifier"), while $\bigcirc$ ("next"), $\Box$ ("always"), and $\mathcal{U}$ ("until") are *temporal operators*. Like in **CTL**, where every temporal operator has to be preceded by a path quantifier, in **ATL** every temporal operator has to be preceded by a coalition quantifier. Thus, *modal operators* of **ATL** are pairs made up of a coalition quantifier and a temporal operator.

We adopt the usual convention that unary connectives have a stronger binding power than binary ones; when this convention helps disambiguate a formula, we usually omit the parentheses associated with binary connectives.

Formulae of the form $\langle\langle A\rangle\rangle\varphi\mathcal{U}\psi$ and $\neg\langle\langle A\rangle\rangle\Box\varphi$ are called *eventualities*, for the reason explained later on.

2.2 **ATL** semantics

While the syntax of **ATL** remained unchanged from [Alur et al. 1997] to [Alur et al. 2002], the semantics, originally based on "alternating transition systems", was revised in [Alur et al. 2002], where the notion of "concurrent game structures" was introduced. The latter are essentially equivalent to "multi-player game models" ([Pauly 2001a], [Pauly 2002]) and are more general than, yet yielding the same set of validities as, alternating transition systems—see [Goranko 2001],[Goranko and Jamroga 2004].

In the present paper, we use the term "concurrent game models" to refer to the "concurrent game structures" from [Alur et al. 2002] and, in keeping with the long-established tradition in modal logic, the term "concurrent game frames" to refer to the structures resulting from those by abstracting away from the meaning of atomic propositions.

2.2.1 *Concurrent game frames.* Concurrent game frames are to **ATL** what Kripke frames are to standard modal logics.

Definition 2.2. A *concurrent game frame* (for short, CGF) is a tuple $\mathfrak{F} = (\Sigma, S, d, \delta)$, where

---

[2]A formal proof of this claim would require a suitable semantic argument, e.g., one involving bisimulations between models for **ATL**. As such an argument would take up quite a lot of space and is not immediately relevant to the contents of the present paper, we do not pursue it here.

— Σ is a finite, non-empty set of *agents*, referred to by the numbers 1 through $|\Sigma|$; subsets of Σ are called *coalitions*;

— $S \neq \emptyset$ is a set of *states*;

— d is a function assigning to every agent $a \in \Sigma$ and every state $s \in S$ a natural number $d_a(s) \geq 1$ of *moves*, or actions, available to agent a at state s; these moves are identified with the numbers 0 through $d_a(s) - 1$. For every state $s \in S$, a *move vector* is a k-tuple $(\sigma_1, \ldots, \sigma_k)$, where $k = |\Sigma|$, such that $0 \leq \sigma_a < d_a(s)$ for every $1 \leq a \leq k$ (thus, σ_a denotes an arbitrary action of agent $a \in \Sigma$). Given a state $s \in S$, we denote by $D_a(s)$ the set $\{0, \ldots, d_a(s) - 1\}$ of all moves available to agent a at s, and by $D(s)$ the set $\prod_{a \in \Sigma} D_a(s)$ of all move vectors at s; with σ we denote an arbitrary member of $D(s)$.

— δ is a *transition function* assigning to every $s \in S$ and $\sigma \in D(s)$ a state $\delta(s, \sigma) \in S$ that results from s if every agent $a \in \Sigma$ plays move σ_a.

All definitions in the remainder of this section refer to an arbitrarily fixed CGF.

Definition 2.3. For two states $s, s' \in S$, we say that s' is a *successor* of s (or, for brevity, an s-successor) if $s' = \delta(s, \sigma)$ for some $\sigma \in D(s)$.

Definition 2.4. A *run* in $\mathfrak{F}$ is an infinite sequence $\lambda = s_0, s_1, \ldots$ of elements of S such that, for all $i \geq 0$, the state s_{i+1} is a successor of the state s_i. Elements of the domain of λ are called *positions*. For a run λ and positions $i, j \geq 0$, we use $\lambda[i]$ and $\lambda[j, i]$ to denote the ith state of λ and the finite segment $s_j, s_{j+1} \ldots, s_i$ of λ, respectively. A run with $\lambda[0] = s$ is referred to as an *s-run*.

Given a tuple τ, we interchangeably use τ_n and $\tau(n)$ to refer to the nth element of τ. We use the symbol $\sharp$ as a placeholder for an arbitrarily fixed move of a given agent.

Definition 2.5. Let $s \in S$ and let $A \subseteq \Sigma$ be a coalition of agents, where $|\Sigma| = k$. An *A-move* σ_A at state s is a k-tuple σ_A such that $\sigma_A(a) \in D_a(s)$ for every $a \in A$ and $\sigma_A(a') = \sharp$ for every $a' \notin A$. We denote by $D_A(s)$ the set of all A-moves at state s.

Alternatively, A-moves at s can be defined as equivalence classes on the set of all move vectors at s, where each equivalence class is determined by the choices of moves of agents in A.

Definition 2.6. We say that a *move vector* σ *extends an A-move* σ_A and write $\sigma_A \sqsubseteq \sigma$, or $\sigma \sqsupseteq \sigma_A$, if $\sigma(a) = \sigma_A(a)$ for every $a \in A$.

Given a coalition $A \subseteq \Sigma$, an A-move $\sigma_A \in D_A(s)$, and a $(\Sigma \setminus A)$-move $\sigma_{\Sigma \setminus A} \in D_{\Sigma \setminus A}(s)$, we denote by $\sigma_A \sqcup \sigma_{\Sigma \setminus A}$ the unique $\sigma \in D(s)$ such that both $\sigma_A \sqsubseteq \sigma$ and $\sigma_{\Sigma \setminus A} \sqsubseteq \sigma$.

Definition 2.7. Let $\sigma_A \in D_A(s)$. The *outcome* of σ_A at s, denoted by $out(s, \sigma_A)$, is the set of all states s' for which there exists a move vector $\sigma \in D(s)$ such that $\sigma_A \sqsubseteq \sigma$ and $\delta(s, \sigma) = s'$.

Concurrent game frames are meant to model coalitions of agents behaving strategically in pursuit of their goals. Given a coalition A, a strategy for A is, intuitively, a rule determining at a given state what A-move the agents in A should play. Given a state as a component of a run, the strategy for agents in A at that state may depend on some part of the history of the run[3], the length of this "remembered" history being a parameter formally represented by an ordinal $\gamma \leq \omega$. Intuitively, players using a γ-recall strategy can "remember" any number $n < \gamma$ of the previous *consecutive* states of the run. If γ is a natural number, then γ can be thought of as a number of the consecutive states, including the current state, on which an agent is basing its decision of what move to play. If, however, $\gamma = \omega$, then an agent can remember any number of the previous consecutive states of the run.

Given a natural number n, by S^n we denote the set of sequences of elements of S of length n; the length of a sequence κ is denoted by $|\kappa|$ and the last element of κ by $l(\kappa)$.

Definition 2.8. Let $A \subseteq \Sigma$ be a coalition and γ an ordinal such that $1 \leq \gamma \leq \omega$. A *$\gamma$-recall strategy for A* (or, a *γ-recall A-strategy*) is a mapping $F_A[\gamma] : \bigcup_{1 \leq n < 1+\gamma} S^n \mapsto \bigcup \{ D_A(s) \mid s \in S \}$ such that $F_A[\gamma](\kappa) \in D_A(l(\kappa))$ for every $\kappa \in \bigcup_{1 \leq n < 1+\gamma} S^n$.

Remark 2.9. Given that $1 + \omega = \omega$, the condition of Definition 2.8 for the case of ω-recall strategies can be rephrased in a simpler form as follows: $F_A[\omega] : \bigcup_{1 \leq n < \omega} S^n \mapsto \bigcup \{ D_A(s) \mid s \in S \}$ such that $F_A[\omega](\kappa) \in D_A(l(\kappa))$ for every $\kappa \in \bigcup_{1 \leq n < \omega} S^n$.

Definition 2.10. Let $F_A[\gamma]$ be a γ-recall A-strategy. If $\gamma = \omega$, then $F_A[\gamma]$ is referred to as a *perfect-recall A-strategy*; otherwise, $F_A[\gamma]$ is referred to as a *bounded-recall A-strategy*. Furthermore, if $\gamma = 1$, then $F_A[\gamma]$ is referred to as a *positional A-strategy*.

Thus, agents using a perfect-recall strategy have potentially unlimited memory; those using positional strategies have none ($\gamma = 1$ means that an agent bases its decisions on one state only, i.e., the current one); in between, agents using n-recall strategies, for $1 < n < \omega$, can base their decisions on the $n-1$ previous consecutive states of the run as well as the current state. We usually write F_A instead of $F_A[\gamma]$ when γ is understood from the context.

Remark 2.11. Even though the concept of n-recall strategies, for $1 < n < \omega$ is of some interest in itself, in the present paper it is introduced for purely technical reasons, to be used in the proof of the satisfiability-wise equivalence (see Theorem 3.9 below) of the semantics of **ATL** based on concurrent game models and the one based on concurrent game Hintikka structures, as well as in the completeness proof for our tableau procedure.

We note, however, that a more realistic notion of finite-memory strategy is the one allowing a strategy to be computed by a finite automaton reading a sequence of states in the history of a run and producing a move to be played, as proposed in [Thomas 1995].

[3]In general, we might consider the case when an agent can remember *any* part of the history of the run; it suffices, however, for our purposes in this paper to consider only those parts that are made up of consecutive states of a run.

Definition 2.12. Let $F_A[\gamma]$ be an A-strategy. The *outcome of* $F_A[\gamma]$ *at state* s, denoted by $out(s, F_A[\gamma])$, is the set of all s-runs λ such that

$$(\gamma)\ \ \lambda[i+1] \in out(\lambda[i], F_A[\gamma](\lambda[j,i])) \text{ holds for all } i \geq 0, \\ \text{where } j = \max(i - \gamma + 1, 0).$$

Note that for positional strategies condition (γ) reduces to

$$(\mathbf{P})\ \ \lambda[i+1] \in out(\lambda[i], F_A(\lambda[i])), \text{ for all } i \geq 0,$$

whereas for perfect-recall strategies it reduces to

$$(\mathbf{PR})\ \ \lambda[i+1] \in out(\lambda[i], F_A(\lambda[0,i])), \text{ for all } i \geq 0.$$

2.2.2 *Truth of **ATL**-formulae.* We are now ready to define the truth of **ATL**-formulae in terms of concurrent game models and perfect-recall strategies.

Definition 2.13. A *concurrent game model* (for short, CGM) is a tuple $\mathcal{M} = (\mathfrak{F}, \mathtt{AP}, L)$, where

— $\mathfrak{F}$ is a concurrent game frame;

— $\mathtt{AP}$ is a set of atomic propositions;

— L is a labeling function $L : S \to \mathcal{P}(\mathtt{AP})$. Intuitively, the set $L(s)$ contains the atomic propositions that are true at state s.

Definition 2.14. Let $\mathcal{M} = (\Sigma, S, d, \delta, \mathtt{AP}, L)$ be a concurrent game model. The satisfaction relation $\Vdash$ is inductively defined for all $s \in S$ and all **ATL**-formulae as follows:

— $\mathcal{M}, s \Vdash p$ iff $p \in L(s)$, for all $p \in \mathtt{AP}$;

— $\mathcal{M}, s \Vdash \neg\varphi$ iff $\mathcal{M}, s \nVdash \varphi$;

— $\mathcal{M}, s \Vdash \varphi \to \psi$ iff $\mathcal{M}, s \Vdash \varphi$ implies $\mathcal{M}, s \Vdash \psi$;

— $\mathcal{M}, s \Vdash \langle\langle A \rangle\rangle \bigcirc \varphi$ iff there exists an A-move $\sigma_A \in D_A(s)$ such that $\mathcal{M}, s' \Vdash \varphi$ for all $s' \in out(s, \sigma_A)$;

— $\mathcal{M}, s \Vdash \langle\langle A \rangle\rangle \Box \varphi$ iff there exists a perfect-recall A-strategy F_A such that $\mathcal{M}, \lambda[i] \Vdash \varphi$ holds for all $\lambda \in out(s, F_A)$ and all positions $i \geq 0$;

— $\mathcal{M}, s \Vdash \langle\langle A \rangle\rangle \varphi \mathcal{U} \psi$ iff there exists a perfect-recall A-strategy F_A such that, for all $\lambda \in out(s, F_A)$, there exists a position $i \geq 0$ with $\mathcal{M}, \lambda[i] \Vdash \psi$ and $\mathcal{M}, \lambda[j] \Vdash \varphi$ holds for all positions $0 \leq j < i$.

Definition 2.15. Let θ be an **ATL**-formula and Γ be a set of **ATL**-formulae.

— θ is *true at a state* s *of a CGM* $\mathcal{M}$ if $\mathcal{M}, s \Vdash \theta$; Γ is *true at* s, denoted $\mathcal{M}, s \Vdash \Gamma$, if $\mathcal{M}, s \Vdash \varphi$ holds for every $\varphi \in \Gamma$;

— θ is *satisfiable in a CGM* $\mathcal{M}$ if $\mathcal{M}, s \Vdash \theta$ holds for some $s \in \mathcal{M}$; Γ is *satisfiable in* $\mathcal{M}$ if $\mathcal{M}, s \Vdash \Gamma$ holds for some $s \in \mathcal{M}$;

— θ is *true in a CGM* $\mathcal{M}$ if $\mathcal{M}, s \Vdash \theta$ holds for every $s \in \mathcal{M}$.

As the clauses for the modal operators $\langle\langle A \rangle\rangle \Box$ and $\langle\langle A \rangle\rangle \mathcal{U}$ in Definition 2.14 involve strategies, these will henceforth be referred to as *strategic operators*.

Remark 2.16. As in the present paper we are only concerned with satisfiability of single formulae (or, equivalently, finite sets of formulae), and a formula can only contain finitely many atomic propositions, the size of AP is of no real significance for our purposes here. The issue of the cardinality of the set of agents Σ is more involved, however, as infinite coalitions can be *named* within a single formula, which would imply certain technical complications. Nevertheless, when interested in satisfiability of single formulae, the finiteness of Σ does not result in a loss of generality. Indeed, as every formula θ mentions only *finitely many coalitions*, we can definite an equivalence relation of finite index on the set of agents that is naturally induced by φ; to wit, two agents are considered "equivalent" if they always occur, or otherwise, together in all the coalitions mentioned in θ (i.e. $a \cong_\theta b$ if $a \in A$ iff $b \in A$ holds for every coalition A mentioned in θ). Then, θ can be rewritten into a formula θ' in which equivalence classes with respect to $\cong_\theta$ are treated as single agents. It is not hard to show that θ' is satisfiable iff θ is, and thus the satisfiability of the latter can be reduced to the satisfiablity of the former.

2.3 Fixpoint characterization of strategic operators

In the tableau procedure described later on in the paper and in the proofs of a number of results concerning **ATL**, we will make use of the fact that the strategic operators $\langle\langle A\rangle\rangle\Box$ and $\langle\langle A\rangle\rangle\mathcal{U}$ can be given neat fixpoint characterizations, as shown in [Goranko and van Drimmelen 2006]. In this respect, **ATL** turns out to be not much different from **LTL** and **CTL**, whose "long-term" modalities are well-known to have similar fixpoint characterizations.

The following definitions introduce set theoretic operators corresponding to the semantics of the respective coalitional modalities in a sense made precise in Theorem 2.19.

Definition 2.17. Let (Σ, S, d, δ) be a CGF and let $X \subseteq S$. Then, $[\langle\langle A\rangle\rangle\bigcirc]$ is an operator $\mathcal{P}(S) \mapsto \mathcal{P}(S)$ defined by the following condition: $s \in [\langle\langle A\rangle\rangle\bigcirc](X)$ iff there exists $\sigma_A \in D_A(s)$ such that $out(s, \sigma_A) \subseteq X$.

Definition 2.18. Let (Σ, S, d, δ) be a CGF and let $X, Y \subseteq S$. Then, we define operators $[Y \cap \langle\langle A\rangle\rangle\bigcirc]$ and $[Y \cup \langle\langle A\rangle\rangle\bigcirc]$ from $\mathcal{P}(S)$ to $\mathcal{P}(S)$ as expected:

— $[Y \cap \langle\langle A\rangle\rangle\bigcirc](X) = Y \cap [\langle\langle A\rangle\rangle\bigcirc](X)$;
— $[Y \cup \langle\langle A\rangle\rangle\bigcirc](X) = Y \cup [\langle\langle A\rangle\rangle\bigcirc](X)$.

Given a formula φ and a model $\mathcal{M}$, we denote by $\|\varphi\|_\mathcal{M}$ the set $\{ s \mid \mathcal{M}, s \Vdash \varphi \}$; we simply write $\|\varphi\|$ when $\mathcal{M}$ is clear from the context.

Given a monotone operator $[\Omega] : \mathcal{P}(S) \mapsto \mathcal{P}(S)$, we denote by $\mu X.[\Omega](X)$ and $\nu X.[\Omega](X)$ the least and greatest fixpoints of $[\Omega]$, respectively.

Theorem 2.19 [Goranko and van Drimmelen 2006]. *Let* $(\Sigma, S, d, \delta, AP, L)$ *be a CGM. Then, for any formulae* φ, ψ*:*

— $\|\langle\langle A\rangle\rangle\bigcirc\varphi\| = [\langle\langle A\rangle\rangle\bigcirc](\|\varphi\|)$
— $\|\langle\langle A\rangle\rangle\Box\varphi\| = \nu X.[\|\varphi\| \cap \langle\langle A\rangle\rangle\bigcirc](X)$;
— $\|\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi\| = \mu X.[\|\psi\| \cup [\|\varphi\| \cap \langle\langle A\rangle\rangle\bigcirc]](X)$.

Corollary 2.20. *The following equivalences hold at every state of every CGM with $A \subseteq \Sigma$:*

— $\langle\langle A\rangle\rangle\Box\varphi \leftrightarrow \varphi \wedge \langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\Box\varphi$;
— $\langle\langle A\rangle\rangle\varphi\mathcal{U}\psi \leftrightarrow \psi \vee (\varphi \wedge \langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\varphi\mathcal{U}\psi)$;

2.4 Tight, general, and loose **ATL**-satisfiability

Unlike the case of standard modal logics, it is natural to think of several apparently different notions of **ATL**-satisfiability. The differences lie along two dimensions: the types of strategies used in the definition of the satisfaction relation and the relationship between the set of agents mentioned in a formula and the set of agents referred to in the language. We consider these issues in turn.

The notion of strategy, as introduced above, is dependent on the amount of memory used to prescribe it. At one end of the spectrum are *positional* (or *memoryless*) strategies, which are based on the current state, but not any part of the history of, the run; and at the other—*perfect recall* strategies, which are based on the entire history of the run. It turns out, however, that these both "extreme" types of strategy—and, hence, all those in between—yield equivalent semantics in the case of **ATL** (they, however, differ in the case of the more expressive logic **ATL***, considered in [Alur et al. 2002]). Therefore, the above definition of truth of **ATL**-formulae (Definition 2.14) could have been couched in terms of positional, rather than perfect-recall, strategies without any changes in what formulae are satisfiable at which states. This equivalence, first mentioned in [Alur et al. 2002], can be proved using a model-theoretic argument; independently, it follows as a corollary of the soundness and completeness theorems for the tableau procedure presented below (see Corollary 5.40).

Now, assuming the type of strategies being fixed, one can consider three different, at least on the face of it, notions of satisfiability and validity for **ATL**, depending on the relationship between the set of agents mentioned in a formula and the set of agents referred to in the language, as introduced in [Walther et al. 2006].

For every **ATL**-formula θ, we denote by Σ_θ the set of agents mentioned in θ. When considering an **ATL**-formula θ in isolation, we may assume, without a loss of generality, that the names of the agents referred to in θ are the numbers 1 through $|\Sigma_\theta|$; hence, the following definitions.

Definition 2.21. An **ATL**-formula θ is Σ-*satisfiable*, for some $\Sigma \supseteq \Sigma_\theta$, if θ is satisfiable in a CGM $\mathcal{M} = (\Sigma, S, d, \delta, \mathtt{AP}, L)$; θ is Σ-*valid* if θ is true in every such CGM.

Definition 2.22. An **ATL**-formula θ is *tightly satisfiable* if θ is satisfiable in a CGM $\mathcal{M} = (\Sigma_\theta, S, d, \delta, \mathtt{AP}, L)$; θ is *tightly valid* if θ is true in every such CGM.

Clearly, θ is tightly satisfiable iff it is Σ_θ-satisfiable.

Definition 2.23. An **ATL**-formula θ is *generally satisfiable* if θ is satisfiable in a CGM $\mathcal{M} = (\Sigma', S, d, \delta, \mathtt{AP}, L)$ for some Σ' with $\Sigma_\theta \subseteq \Sigma'$; θ is *generally valid* if θ is true in every such CGM.

To see that tight satisfiability (validity) is different from general satisfiability (validity), consider the formula $\neg\langle\langle 1\rangle\rangle\bigcirc p \wedge \neg\langle\langle 1\rangle\rangle\bigcirc\neg p$; it is easy to see that this formula is generally, but not tightly satisfiable (accordingly, its negation is tightly, but not generally, valid). Obviously, tight satisfiability implies general satisfiability, and it is not hard to notice that it also implies Σ-satisfiability for any $\Sigma \supset \Sigma_\theta$ (in a model where any agent $a' \in \Sigma \setminus \Sigma_\theta$ plays a dummy role by having exactly one action available at every state).

We now show that testing for both Σ-satisfiability and general satisfiability for θ can be reduced to testing for tight satisfiability and a special case of Σ-satisfiability where $\Sigma = \Sigma_\theta \cup \{a'\}$ for some $a' \notin \Sigma_\theta$ (more precisely, $a' = |\Sigma_\theta| + 1$)—in other words, only one new agent suffices to witness satisfiability of θ over CGFs involving agents not in Σ_θ. This result, proved below, was first stated, with a proof sketch, for satisfiability in the more restricted (but equivalent with respect to satisfiability, see [Goranko 2001]) semantics based on "alternating transition systems", in [Walther et al. 2006].

Theorem 2.24. *Let θ be an **ATL**-formula, $\Sigma_\theta \subsetneq \Sigma$, and $a' \notin \Sigma_\theta$. Then, θ is Σ-satisfiable iff θ is $(\Sigma_\theta \cup \{a'\})$-satisfiable.*

Proof. Suppose, first, that θ is Σ-satisfiable. Let $\mathcal{M} = (\Sigma, S, d, \delta, \mathtt{AP}, L)$ be a CGM and $s \in S$ be a state such that $\mathcal{M}, s \Vdash \theta$. To obtain a $(\Sigma_\theta \cup \{a'\})$-model $\mathcal{M}'$ for θ, first, let, for every $s \in S$:

— $d'_a(s) = d_a(s)$ for every $a \in \Sigma_\theta$;
— $d'_{a'}(s) = |\prod_{b\in(\Sigma-\Sigma_\theta)} d_b(s)|$;

then, define δ' in the following way: $\delta'(\sigma_{\Sigma_\theta} \sqcup \sigma_{a'}) = \delta(\sigma_{\Sigma_\theta} \sqcup \sigma_{\Sigma-\Sigma_\theta})$, where $\sigma_{a'}$ is the place of $\sigma_{\Sigma-\Sigma_\theta}$ in the lexicographic ordering of $D_{\Sigma-\Sigma_\theta}(s)$. Finally, put $\mathcal{M}' = (\Sigma_\theta \cup \{a'\}, S, d', \delta', \mathtt{AP}, L)$.

Notice that the above definition immediately implies that $out(s, \sigma_A)$ is the same set in both $\mathcal{M}$ and $\mathcal{M}'$ for every $s \in S$ and every $\sigma_A \in D_A(s)$ with $A \subseteq \Sigma_\theta$, and therefore, in both models, $[\langle\langle A\rangle\rangle\bigcirc](X)$ is the same set for every $X \subseteq S$ and every $A \subseteq \Sigma_\theta$. It can then be shown, by a routine induction on the structure of subformulae χ of θ, using Theorem 2.19, that $\mathcal{M}, s \Vdash \chi$ iff $\mathcal{M}', s \Vdash \chi$ for every $s \in S$.

Suppose, next, that θ is $(\Sigma_\theta \cup \{a'\})$-satisfiable. Let $\mathcal{M}$ be the model witnessing the satisfaction and let b be an arbitrary agent in $\Sigma - \Sigma_\theta$. To obtain a Σ-model $\mathcal{M}'$ for θ, first, let, for every $s \in S$:

— $d'_a(s) = d_a(s)$ for every $a \in \Sigma_\theta$;
— $d'_b(s) = d_{a'}(s)$;
— $d'_{b'}(s) = 1$ for any $b' \in \Sigma \setminus (\{b\} \cup \Sigma_\theta)$;

then, define δ' in the following way: $\delta'(\sigma_{\Sigma_\theta} \sqcup \sigma_{\Sigma-\Sigma_\theta}) = \delta(\sigma_{\Sigma_\theta} \sqcup \sigma_{a'})$, where $\sigma_{a'} = \sigma_b$. Finally, put $\mathcal{M}' = (\Sigma, S, d', \delta', \mathtt{AP}, L)$. The rest of the argument is identical to the one for the opposite direction. □

Corollary 2.25. *Let θ be an **ATL**-formula. Then, θ is generally satisfiable iff θ is either tightly satisfiable or $(\Sigma_\theta \cup \{a'\})$-satisfiable for any $a' \notin \Sigma_\theta$.*

Proof. Straightforward. □

Theorem 2.24 and Corollary 2.25 essentially mean that it suffices to consider two distinct notions of satisfiability for **ATL**-formulae: tight satisfiability and satisfiability in CGMs with one fresh agent, which we will henceforth refer to as *loose satisfiability.*

2.5 Alternative semantic characterization of negated modal operators

Under Definition 2.14, truth conditions for negated modal operators, such as $\neg\langle\langle A\rangle\rangle\,\mathcal{U}$, involve claims about the non-existence of moves or strategies. In [Goranko and van Drimmelen 2006], an alternative semantic characterization of such formulae has been proposed; this alternative characterization involves claims about the existence of the so-called in [Goranko and van Drimmelen 2006] *co-moves* and *co-strategies.*

Definition 2.26. Let $s \in S$ and $A \subseteq \Sigma$. A *co-A-move* at state s is a function $\sigma^c_A : D_A(s) \mapsto D(s)$ such that $\sigma_A \sqsubseteq \sigma^c_A(\sigma_A)$ for every $\sigma_A \in D_A(s)$. We denote the set of all co-A-moves at s by $D^c_A(s)$.

Intuitively, given an A-move $\sigma_A \in D_A(s)$, which represents a collective action of agents in A, a co-A-move assigns to σ_A a "countermove" $\sigma_{\Sigma\setminus A}$ of the complement coalition $\Sigma \setminus A$; taken together, these two moves produce a unique move vector $\sigma_A \sqcup \sigma_{\Sigma\setminus A} \in D(s)$.

Definition 2.27. Let $\sigma^c_A \in D^c_A(s)$. The outcome of σ^c_A at s, denoted by $out(s, \sigma^c_A)$, is the set $\bigcup\{\,\delta(s, \sigma^c_A(\sigma_A)) \mid \sigma_A \in D_A(s)\,\}$. (Thus, $out(s, \sigma^c_A)$ is the range of σ^c_A).

We next define co-strategies, which are related to co-moves in the same way as strategies are related to moves.

Definition 2.28. Let $A \subseteq \Sigma$ be a coalition and γ an ordinal such that $1 \leq \gamma \leq \omega$. A *$\gamma$-recall co-A-strategy* is a mapping $F^c_A[\gamma] : \bigcup_{\,1\leq n<1+\gamma} S^n \mapsto \bigcup\{\,D^c_A(s) \mid s \in S\,\}$ such that $F^c_A[\gamma](\kappa) \in D^c_A(l(\kappa))$ for every $\kappa \in \bigcup_{\,1\leq n<1+\gamma} S^n$.

Note that the coalition following a co-A-strategy is $\Sigma \setminus A$.

Remark 2.29. Given that $1 + \omega = \omega$, the condition of the Definition 2.28 for the case of ω-recall strategies can be rephrased in a simpler form as follows: $F^c_A[\omega] : \bigcup_{\,1\leq n<\omega} S^n \mapsto \bigcup\{\,D^c_A(s) \mid s \in S\,\}$ such that $F^c_A[\omega](\kappa) \in D^c_A(l(\kappa))$ for every $\kappa \in \bigcup_{\,1\leq n<\omega} S^n$.

Remark 2.30. A γ-recall co-strategy can be defined equivalently as a mapping from pairs ($\kappa \in S^n$; γ-recall strategy $F_A[\gamma]$) to the set of outcome states $out(l(\kappa), F_A[\gamma](\kappa))$.

We will write F^c_A instead of $F^c_A[\gamma]$ when γ is understood from the context.

Definition 2.31. Let $F^c_A[\gamma]$ be a γ-recall co-A-strategy. If $\gamma = \omega$, then $F^c_A[\gamma]$ is referred to as a *perfect-recall co-A-strategy*; otherwise, $F_A[\gamma]$ is referred to as a *bounded-recall co-A-strategy.* Furthermore, if $\gamma = 1$, then $F^c_A[\gamma]$ is referred to as a *positional co-A-strategy.*

Definition 2.32. Let $F^c_A[\gamma]$ be a co-A-strategy. The *outcome of $F^c_A[\gamma]$ at state* s, denoted by $out(s, F^c_A[\gamma])$, is the set of all s-runs λ such that

$$(\gamma^c)\quad \lambda[i+1] \in out(\lambda[i], F^c_A(\lambda[j,i])) \text{ holds for all } i \geq 0, \text{ where } j = \max(i-\gamma+1, 0).$$

For positional co-strategies, condition ($\gamma^{\mathbf{c}}$) reduces to

$$(\mathbf{CP})\ \ \lambda[i+1] \in out(\lambda[i], F^{c}_{A}(\lambda[i])),\ \text{ for all } i \geq 0,$$

whereas for perfect-recall co-strategies, it reduces to

$$(\mathbf{CPR})\ \ \lambda[i+1] \in out(\lambda[i], F^{c}_{A}(\lambda[0,i])),\ \text{ for all } i \geq 0.$$

Now, we can give alternative truth conditions for negated modalities, couched in terms of co-moves and co-strategies.

Theorem 2.33 [Goranko and van Drimmelen 2006]. *Let $\mathcal{M}$ be a CGM and $s \in \mathcal{M}$. Then,*

— *$\mathcal{M}, s \Vdash \neg\langle\langle A\rangle\rangle \bigcirc \varphi$ iff there exists a co-A-move $\sigma^{c}_{A} \in D^{c}_{A}(s)$ such that $\mathcal{M}, s' \Vdash \neg\varphi$ for every $s' \in out(s, \sigma^{c}_{A})$;*

— *$\mathcal{M}, s \Vdash \neg\langle\langle A\rangle\rangle \Box \varphi$ iff there exists a perfect recall co-A-strategy F^{c}_{A} such that, for every $\lambda \in out(s, F^{c}_{A})$, there exists position $i \geq 0$ with $\mathcal{M}, \lambda[i] \Vdash \neg\varphi$;*

— *$\mathcal{M}, s \Vdash \neg\langle\langle A\rangle\rangle \varphi \mathcal{U} \psi$ iff there exists a perfect recall co-A-strategy F^{c}_{A} such that, for every $\lambda \in out(s, F^{c}_{A})$ and every position $i \geq 0$ with $\mathcal{M}, \lambda[i] \Vdash \psi$, there exists a position $0 \leq j < i$ with $\mathcal{M}, \lambda[j] \Vdash \neg\varphi$.*

Remark 2.34. Since both types of strategies yield the same semantics for **ATL**, in the last two clauses of Theorem 2.33, "perfect recall" can be replaced with "positional".

3. HINTIKKA STRUCTURES FOR **ATL**

When proving completeness of the tableau procedure described in the next section, we will make use of a new kind of semantic structures for **ATL**—namely, Hintikka structures. The basic difference between models and Hintikka structures is that while models specify the truth or otherwise of every formula of the language at every state, Hintikka structures only provide truth values of the formulae relevant to the evaluation of a fixed formula θ. Before defining Hintikka structures for **ATL**, which we, for the sake of terminological consistency, call *concurrent game Hintikka structures*, we introduce, with a view to simplifying the subsequent presentation, α- and β-notation for **ATL**-formulae.

3.1 α- and β-notation for **ATL**

We divide all **ATL**-formulae into primitive and non-primitive ones.

Definition 3.1. Let φ be an **ATL**-formula. Then, φ is *primitive* if it is one of the following:

— $\top$;
— $p \in \mathtt{AP}$;
— $\neg p$ for some $p \in \mathtt{AP}$;
— $\langle\langle A\rangle\rangle \bigcirc \psi$ for some formula ψ;
— $\neg\langle\langle A\rangle\rangle \bigcirc \psi$ for some formula ψ and $A \neq \Sigma$.

Otherwise, φ is *non-primitive.*

Intuitively, φ is primitive if the truth of φ at a state s of a CGM cannot be reduced to the truth of any "semantically simpler" formulae at s; otherwise, φ is non-primitive. Note, in particular, that $\neg p$ is not considered "semantically simpler" then p, as the truth of the former can not be reduced to the truth, as opposed to the falsehood, of the latter.

Following [Smullyan 1968], we classify all non-primitive formulae into α-ones and β-ones. Intuitively, α-formulae are "conjunctive" formulae: an α-formula is true at a state s iff two other formulae, "conjuncts" of α, denoted by α_1 and α_2, are true at s. By contrast, β-formulae are "disjunctive" formulae, true at a state s iff either of their "disjuncts", denoted by β_1 and β_2, is true at s. For neatness of classification, if the truth of a non-primitive formula ψ at s can be reduced to the truth of only *one* simpler formula at s, then ψ is treated as an α-formula; for such formulae, $\alpha_1 = \alpha_2$. The following tables list α- and β-formulae together with their respective "conjuncts" and "disjuncts".

α	α_1	α_2
$\neg\neg\varphi$	φ	φ
$\neg(\varphi \to \psi)$	φ	$\neg\psi$
$\neg\langle\langle\Sigma\rangle\rangle\bigcirc\varphi$	$\langle\langle\emptyset\rangle\rangle\bigcirc\neg\varphi$	$\langle\langle\emptyset\rangle\rangle\bigcirc\neg\varphi$
$\langle\langle A\rangle\rangle\Box\varphi$	φ	$\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\Box\varphi$

β	β_1	β_2
$\varphi \to \psi$	$\neg\varphi$	ψ
$\langle\langle A\rangle\rangle(\varphi\,\mathcal{U}\psi)$	ψ	$\varphi \wedge \langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle(\varphi\,\mathcal{U}\psi)$
$\neg\langle\langle A\rangle\rangle(\varphi\,\mathcal{U}\psi)$	$\neg\psi \wedge \neg\varphi$	$\neg\psi \wedge \neg\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle(\varphi\,\mathcal{U}\psi)$
$\neg\langle\langle A\rangle\rangle\Box\varphi$	$\neg\varphi$	$\neg\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\Box\varphi$

The entries for the non-modal connectives in the above tables are motivated by the well-known classical validities. The entries for the strategic operators are motivated by Corollary 2.20. Lastly, it can be easily checked that $\mathcal{M}, s \Vdash \neg\langle\langle\Sigma\rangle\rangle\bigcirc\varphi$ iff $\mathcal{M}, s \Vdash \langle\langle\emptyset\rangle\rangle\bigcirc\neg\varphi$ for every CGM $\mathcal{M}$ and $s \in \mathcal{M}$.

3.2 Concurrent game Hintikka structures

We are now ready to define concurrent game Hintikka structures. Like concurrent game models, concurrent game Hintikka structures are based on concurrent game frames, where different kinds of strategies may be used, ranging from positional to perfect-recall. As it will become evident from the forthcoming completeness proof of our tableau procedure, in the case of basic **ATL**, which we primarily focus on in this paper, it suffices to consider only *positional* Hintikka structures. Nevertheless, we consider, in this section, the most general case of Hintikka structures, based on *perfect-recall* strategies[4].

Definition 3.2. A (perfect-recall) *concurrent game Hintikka structure* (for short, CGHS) is a tuple $\mathcal{H} = (\Sigma, S, d, \delta, H)$, where

[4]Our reason for doing so is that we intend to consider, in follow-up work, adaptations of the tableau procedure described herein to some important variations and extensions of **ATL**, such as **ATL** with incomplete information, **ATL***, and Game Logic ([Alur et al. 2002]), where positional strategies only do not suffice; then, the results of this section will be put to full use.

— (Σ, S, d, δ) is a concurrent game frame;

— H is a labeling of the elements of S with sets of **ATL**-formulae that satisfy the following constraints:

H1. If $\neg\varphi \in H(s)$, then $\varphi \notin H(s)$;
H2. if $\alpha \in H(s)$, then $\alpha_1 \in H(s)$ and $\alpha_2 \in H(s)$;
H3. if $\beta \in H(s)$, then $\beta_1 \in H(s)$ or $\beta_2 \in H(s)$;
H4. if $\langle\langle A\rangle\rangle \bigcirc \varphi \in H(s)$, then there exists an A-move $\sigma_A \in D_A(s)$ such that $\varphi \in H(s')$ for all $s' \in out(s, \sigma_A)$;
H5. if $\neg\langle\langle A\rangle\rangle \bigcirc \varphi \in H(s)$, then there exists a co-A-move $\sigma^c_A \in D^c_A(s)$ such that $\neg\varphi \in H(s')$ for all $s' \in out(s, \sigma^c_A)$;
H6. if $\langle\langle A\rangle\rangle \varphi \mathcal{U} \psi \in H(s)$, then there exists a perfect-recall A-strategy F_A such that, for all $\lambda \in out(s, F_A)$, there exists a position $i \geq 0$ such that $\psi \in H(\lambda[i])$ and $\varphi \in H(\lambda[j])$ holds for all positions $0 \leq j < i$;
H7. if $\neg\langle\langle A\rangle\rangle \Box \varphi \in H(s)$, then there exists a perfect-recall co-A-strategy F^c_A such that, for every $\lambda \in out(s, F^c_A)$, there exists position $i \geq 0$ with $\neg\varphi \in H(\lambda[i])$.

Remark 3.3. To obtain the definition of positional CGHS, all one has to do is replace "perfect-recall" with "positional" in clauses (H6) and (H7) of Definition 3.2.

Definition 3.4. Let θ be an **ATL**-formula and $\mathcal{H} = (\Sigma, S, d, \delta, H)$ be a CGHS. We say that $\mathcal{H}$ is a *concurrent game Hintikka structure for* θ if $\theta \in H(s)$ for some $s \in S$.

Hintikka structures can be thought of as representing a class of models on the set of states S that, for every $s \in S$, agree on the formulae in $H(s)$—that is, make exactly the same formulae in $H(s)$ true. Models themselves can be thought of as *maximal* Hintikka structures, whose states are labeled with maximally consistent sets of formulae. More precisely, given a CGM $\mathcal{M} = (\Sigma, S, d, \delta, \mathtt{AP}, L)$, we can define the extended labeling function $L^+_{\mathcal{M}}$ by $L^+_{\mathcal{M}}(s) = \{ \varphi \mid \mathcal{M}, s \Vdash \varphi \}$, where φ ranges over all **ATL**-formulae, and the resulting structure $(\Sigma, S, d, \delta, L^+_{\mathcal{M}})$ will be a Hintikka structure. This immediately gives rise to the following theorem.

THEOREM 3.5. *Let θ be an **ATL**-formula. Every CGM $\mathcal{M} = (\Sigma, S, d, \delta, \mathtt{AP}, L)$ satisfying θ induces a CGHS $\mathcal{H} = (\Sigma, S, d, \delta, L^+_{\mathcal{M}})$ for θ, where $L^+_{\mathcal{M}}$ is the extended labeling function on $\mathcal{M}$.*

PROOF. Straightforward, using Theorem 2.33 for (H5) and (H7). □

Conversely, every Hintikka structure for a formula θ can be expanded to a maximal one—that is, a model—by declaring, for every $s \in S$, all atomic propositions outside $H(s)$ to be false at s. To prove this claim, however, we need a few auxiliary definitions.

Definition 3.6. Let $\mathcal{H} = (\Sigma, S, d, \delta, H)$ be a CGHS. A *run* of length m, where $1 \leq m < \omega$, in $\mathcal{H}$ is a sequence $\lambda = s_0, \ldots, s_{m-1}$ of elements of S such that, for all $0 \leq i < m - 1$, the state s_{i+1} is a successor of the state s_i. Numbers 0 through $m - 1$ are called *positions* of λ. The length of λ, defined as the number of positions in λ, is denoted by $|\lambda|$. For each position $0 \leq i < m$, we denote by $\lambda[i]$ the ith state of λ. A *finite run* in $\mathcal{H}$ is a run of length m for some m with $1 \leq m < \omega$. A finite run with $\lambda[0] = s$ is a *finite s-run*.

Definition 3.7. Let $\mathcal{H}$ be a CGHS, λ be a finite s-run in $\mathcal{H}$, and $F^c_A[m]$ be an m-recall co-A-strategy on the frame of $\mathcal{H}$, where $1 \leq m < \omega$. We say that λ is *compliant with* $F^c_A[m]$ if

— $|\lambda| = m + 1$;

— $\lambda[i+1] \in out(\lambda[i], F^c_A[m](\lambda[0,i]))$ holds for all $0 \leq i < m$.

Definition 3.8. Let $\mathcal{H}$ be a CGHS, let λ be an (infinite) s-run in $\mathcal{H}$ and let F^c_A be a perfect-recall co-A-strategy on the frame of $\mathcal{H}$. We say that λ is *compliant with* F^c_A if $\lambda \in out(s, F^c_A)$.

Theorem 3.9. *Let θ be an **ATL**-formula. Every CGHS $\mathcal{H} = (\Sigma, S, d, \delta, H)$ for θ can be expanded to a CGM satisfying θ.*

Proof. Let $\mathcal{H} = (\Sigma, S, d, \delta, H)$ be a CGHS for θ. To obtain a CGM $\mathcal{M} = (\Sigma, S, d, \delta, \mathtt{AP}, L)$, we define the labeling function L as follows: $L(s) = H(s) \cap \mathtt{AP}$, for every $s \in S$.

To establish the statement of the theorem, we prove, by induction on the structure of formula χ that, for every $s \in S$ and every χ, the following claim holds:

$$\chi \in H(s) \text{ implies } \mathcal{M}, s \Vdash \chi \textit{ and } \neg\chi \in H(s) \text{ implies } \mathcal{M}, s \Vdash \neg\chi.$$

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

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

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

The cases of $\chi = \varphi \to \psi$ and $\chi = \langle\!\langle A \rangle\!\rangle \bigcirc \psi$ and are straightforward, using (H2)–(H5).

Suppose that $\chi = \langle\!\langle A \rangle\!\rangle \varphi \mathcal{U} \psi$. If $\langle\!\langle A \rangle\!\rangle \varphi \mathcal{U} \psi \in H(s)$, then the desired conclusion immediately follows from (H6) and the inductive hypothesis.

Assume now that $\neg\langle\!\langle A \rangle\!\rangle \varphi \mathcal{U} \psi \in H(s)$. In view of the inductive hypothesis and Theorem 2.33, it suffices to show that there exists a perfect-recall co-A-strategy F^c_A such that $\lambda \in out(s, F^c_A)$ implies that, if there exists $i \geq 0$ with $\psi \in H(\lambda[i])$, then there exists $0 \leq j < i$ with $\neg\varphi \in H(\lambda[j])$.

We define the required F^c_A by induction on the length of sequences in its domain. This amounts to defining finite prefixes of F^c_A for every $1 \leq n < \omega$—the restrictions of F^c_A to sequences of states of length $\leq n$. Clearly, the finite prefix of F^c_A of length n is an n-recall co-A-strategy. We only explicitly define the value of $F^c_A[n](\lambda)$, where $|\lambda| = n$, if λ is a finite s-run compliant with $F^c_A[n-1]$ (recall Definition 3.7), where $F^c_A[n-1]$ is a co-A-strategy defined at the previous step of the induction. The values of $F^c_A[n](\lambda)$ for any other sequences of length n are immaterial. The only other constraint that we have to take into account when defining $F^c_A[n]$ is that, if $F^c_A[n]$ extends $F^c_A[m]$, then the values of $F^c_A[m]$ and $F^c_A[n]$ should agree on all the sequences of length m. Alongside defining $F^c_A[n]$ for every $1 \leq n < \omega$, we prove that the following invariant property holds: If $\lambda \in out(s, F^c_A[n])$, then

$$(\dagger)\quad \begin{array}{ll} (i) & \textit{Either} \text{ there exists a position } 0 \leq i \leq n, \text{ such that} \\ & \neg\varphi \in H(\lambda[i]) \text{ and } \neg\psi \in H(\lambda[j]) \text{ for all } 0 \leq j \leq i, \\ (ii) & \textit{or } \neg\psi, \neg\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi \in H(\lambda[i]) \text{ for all } 0 \leq i \leq n. \end{array}$$

Clearly, if every finite prefix of F^c_A satisfies (†), then F^c_A is the required co-A-strategy.

We start by defining $F^c_A[1]$. There is only one s-run of length 1, namely (s), and compliancy condition is, in this case, vacously true. As $\neg\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi \in H(s)$, in view of (H3) and (H2), either $\neg\psi, \neg\varphi \in H(s)$ or $\neg\psi, \neg\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi \in H(s)$. In the former case any co-A-move will produce a co-A-strategy $F^c_A[1]$ such that, if $\lambda \in out(s, F^c_A[1])$, then λ satisfies (†) (i). In the latter case, (H5) guarantees that there exists a co-A-move $\sigma^c_A \in D^c_A(s)$ such that $\neg\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi \in H(s')$ for all $s' \in out(s, \sigma^c_A)$. This, together with (H3) and (H2) guarantees that $\neg\psi, \neg\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi \in H(s')$ for every $s' \in out(s, \sigma^c_A)$, which, as $\neg\psi \in H(s)$, ensures that (†) (ii) holds for any $\lambda \in out(s, F^c_A[1])$. Thus, in either case, (†) holds for every $\lambda \in out(s, F^c_A[1])$.

Next, inductively assume that, if λ is an s-run compliant with $F^c_A[n]$, then (†) holds for λ. We need to show how to extend $F^c_A[n]$ to $F^c_A[n+1] \supset F^c_A[n]$ in the (†)-preserving way. If (†) (i) holds for every λ satisfying the condition of the inductive hypothesis, then obviously, any co-A-move will do. Otherwise, (†) (ii) holds for every λ satisfying the inductive assumption for which (†) (i) does not hold; then, $F^c_A[n+1]$ can be obtained from $F^c_A[n]$ as in the second part of the "basis case" argument. For all other sequences κ of length $n+1$ (i.e., those that do not start with s or are not compliant with $F^c_A[n]$), the value $F^c_A[n](\kappa)$ can be defined arbitrarily. For all sequences κ of length $\leq n$, we stipulate $F^c_A[n+1](\kappa) = F^c_A[n](\kappa)$. This completes the definition of $F^c_A[n+1]$. As we have seen, if λ is an s-run compliant with $F^c_A[n+1]$, then (†) holds for λ.

The case of $\neg\langle\langle A\rangle\rangle\Box\varphi \in H(s)$ is straightforward using (H7), while the case of $\langle\langle A\rangle\rangle\Box\varphi \in H(s)$ can be proved in a way analogous to the case of $\neg\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$, using suitable definitions of compliancy of (finite and infinite) runs with strategies. □

Theorems 3.5 and 3.9 taken together mean that, from the point of view of a single **ATL**-formula, satisfiability in a (perfect-recall) model and in a (perfect-recall) Hintikka structure are equivalent.

4. TERMINATING TABLEAUX FOR TIGHT **ATL**-SATISFIABILITY

In the current section, we present a tableau method for testing **ATL**-formulae for tight satisfiability.

Traditionally, tableau techniques work by decomposing the formula whose satisfiability is being tested into "semantically simpler" formulae. In the classical propositional case ([Smullyan 1968]), "semantically simpler" implies "smaller", which by itself guarantees termination of the procedure in a finite number of steps. Another feature of the tableau method for the classical propositional logic is that this decomposition into semantically simpler formulae results in a tree representing an exhaustive search for a model—or, to be more precise, a Hintikka set (the classical analogue of Hintikka structures)—for the input formula. If at least one branch of the tree produces a Hintikka set for the input formula, the search has succeeded and the formula is pronounced satisfiable[5].

These two defining features of the classical tableau method do not emerge unscathed when the method is applied to logics containing fixpoint operators, such as **ATL** (in this respect, the case of **ATL** is similar to those of **LTL** and **CTL**).

Firstly, decomposition of **ATL**-formulae into "semantically simpler" ones, which, just as in the classical case, is carried out by breaking up α- and β-formulae into their respective "conjuncts" and "disjuncts," does not always produce smaller formulae, as can be seen from the tables given at the end of section 3.1. Therefore, we will have to take special precautions to ensure that the procedure terminates (in our case, as in [Wolper 1985], this will involve the use of the so-called *prestates*).

Secondly, in the classical case the only reason why it might turn out to be impossible to produce a Hintikka set for the input formula is that every attempt to build such a set results in a collection of formulae containing a patent inconsistency (henceforth, by *patent inconsistency* we mean a pair of formulas of the form φ, $\neg\varphi$)[6]. In the case of **ATL**, there are two other reasons for a tableau not to correspond to any Hintikka structure for the input formula. First, applying decomposition rules to eventualities—formulae whose truth conditions require that some formula (ψ in the case of the eventuality $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$, and $\neg\varphi$ in the case of the eventuality $\neg\langle\langle A\rangle\rangle\Box\varphi$) "eventually" becomes true; the tableau analog of this we will refer to as *realization of an eventuality,*—one can indefinitely postpone their realization by always choosing the "disjunct" (notice that both eventualities are β-formulas) "promising" that the realization will happen further down the line, this "promise" never being fulfilled. Therefore, in addition to not containing patent inconsistencies, "good" **ATL** tableaux should not contain sets with unrealized eventualities. Yet another reason for the resultant tableau not to represent a Hintikka structure is that some sets do not have all the successors they would be required to have in a corresponding Hintikka structure.

Coming back to the realization of eventualities, it should be noted that, in a Hintikka structure for the input formula, all the eventualities belonging to the labels of its states have to be realized, and different eventualities can place different demands on the labels of states of a Hintikka structure. Fortunately, in the case of **ATL** (just like in those of **LTL** and **CTL** and unlike, for example, those of Parikh's game logic [Pauly and Parikh 2003] and propositional μ-calculus [Bradfield and Stirling 2007]), eventualities can be "taken on" one at a time: we can ensure, and this lies at the heart of our completeness proof, that having realized eventualities one by one, we can then assemble a Hintikka structure out of the "building blocks" realizing single eventualities. This technique resembles the mosaic method used to prove decidability of a variety of modal and temporal logics (see, for example, [Marx et al. 2000]).

[5]Even though this tree is usually built step-by-step by decomposing one formula at a time (see [Smullyan 1968] and [Wolper 1985]), it can be compressed into a simple tree—i.e., a tree with a single interior node—whose root is the set containing only the input formula and whose leaves are all minimal downward-saturated extensions (to be defined later on; see Definitions 4.1 and 4.2) of the root. We will use this, more compact, approach in our tableaux.

[6]Notice that this condition implies but is not, in general, equivalent to propositional inconsistency.

4.1 Brief description of the tableau procedure

In essence, the tableau procedure for testing an **ATL**-formula θ for satisfiability is an attempt to construct a non-empty graph $\mathcal{T}^\theta$, called a *tableau*, representing all possible concurrent game Hintikka structures for θ. If the attempt is successful, θ is pronounced satisfiable; otherwise, it is declared unsatisfiable. (As this whole section is exclusively concerned with tight satisfiability, whenever we use the word "satisfiable" or any derivative thereof, we mean the tight variety; another reason to keep the language generic is that—as we shall see later on—the basic ideas transfer smoothly over to other species of satisfiability).

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

During the construction phase, the construction rules are used to produce a directed graph $\mathcal{P}^\theta$—referred to as the *pretableau* for θ—whose set of nodes properly contains the set of nodes of the tableau $\mathcal{T}^\theta$ that we are ultimately building. Nodes of $\mathcal{P}^\theta$ are sets of **ATL**-formulae, some of which—referred to as *states*[7] — are meant to represent states (whence the name) of a Hintikka structure, while others—referred to as *prestates*—fulfill a purely technical role of helping to keep $\mathcal{P}^\theta$ finite. During the prestate elimination phase, we create a smaller graph $\mathcal{T}_0^\theta$ out of $\mathcal{P}^\theta$—referred to as the *initial tableau for* θ—by eliminating all the prestates of $\mathcal{P}^\theta$ (and tweaking with its edges) since prestates have already fulfilled their function: as we are not going to add any more nodes to the graph built so far, the possibility of producing an infinite structure is no longer a concern. Lastly, during the state elimination phase, we remove from $\mathcal{T}_0^\theta$ all the states, if any, that cannot be satisfied in any CGHS, for one of the following three reasons: either they are inconsistent, or contain unrealizable eventualities, or do not have all the successors needed for their satisfaction. This results in a (possibly empty) subgraph $\mathcal{T}^\theta$ of $\mathcal{T}_0^\theta$, called the *final tableau for* θ. Then, if we have some state Δ in $\mathcal{T}^\theta$ containing θ, we pronounce θ satisfiable; otherwise, we declare θ unsatisfiable.

4.2 Construction phase

As already mentioned, at the construction phase, we build the pretableau $\mathcal{P}^\theta$ — a directed graph whose nodes are sets of **ATL**-formulae, coming in two varieties: *states* and *prestates*. Intuitively, states are meant to represent states of CGHSs, while prestates are "embryo states", which will in the course of the construction be "unwound" into states. Technically, states are downward saturated, while prestates do not have to be so.

[7]From here on, the term "state" is used in two different meanings: as "state" of the (pre)tableaux—which is a set of **ATL**-formulas satisfying certain conditions, to be stated shortly,—and as "state" of a semantic structure (frame, model, or Hintikka structure). Usually, the context will make clear which of these we mean; when the context leaves room for ambiguity, we will explicitly mention what kind of states we are talking about.

Definition 4.1. Let Δ be a set of **ATL**-formulae. We say that Δ is *downward saturated* if the following conditions are satisfied:

— if $\alpha \in \Delta$, then $\alpha_1 \in \Delta$ and $\alpha_2 \in \Delta$;
— if $\beta \in \Delta$, then $\beta_1 \in \Delta$ or $\beta_2 \in \Delta$.

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

Another type of edge represents transitions in CGHSs effected by move vectors. Accordingly, this type of edge will be represented in (pre)tableaux by single arrows marked with $|\Sigma_\theta|$-tuples σ of numbers, each number intuitively representing an a-move for some $a \in \Sigma_\theta$. Intuitively, we think of these $|\Sigma_\theta|$-tuples as move vectors. Thus, if we draw an arrow marked by σ from a state Δ to a prestate Γ (depicted as $\Delta \stackrel{\sigma}{\longrightarrow} \Gamma$), this intuitively means that, in any CGHS represented by the tableau we are building, from a state satisfying Δ we can move along σ to a state satisfying Γ.

It should be noted that, in the pretableau, we never create in one go full-fledged successors for states, which is to say we never draw a marked arrow from state to state; such arrows always go from states to prestates. On the other hand, unmarked arrows connect prestates to states. Thus, the whole construction of the pretableau alternates between going from prestates to states along edges represented by double unmarked arrows and going from states to prestates along the edges represented by single arrows marked by "move vectors". This cycle has, however, to start somewhere.

The tableau procedure for testing satisfiability of θ starts off with the creation of a single prestate $\{\theta\}$. Thereafter, a pair of construction rules are applied to the part of the pretableau created thus far: one of the rules, **(SR)**, specifies how to unwind prestates into states; the other, **(Next)**,—how to obtain "successor" prestates from states. To state **(SR)**, we need the following definition.

Definition 4.2. Let Γ and Δ be sets of **ATL**-formulae. We say that Δ is a *minimal downward saturated extension of* Γ if the following holds:

— $\Gamma \subseteq \Delta$;
— Δ is downward saturated;
— there is no downward saturated set Δ' such that $\Gamma \subseteq \Delta' \subset \Delta$.

Note that Γ can be a minimal downward saturated extension of itself.

We now state the first construction rule.

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

(1) add to the pretableau all the minimal downward saturated extensions Δ of Γ as *states*;

(2) for each of the so obtained states Δ, if Δ does not contain any formulae of the form $\langle\!\langle A\rangle\!\rangle \bigcirc \varphi$ or $\neg\langle\!\langle A\rangle\!\rangle \bigcirc \varphi$, add the formula $\langle\!\langle \Sigma_\theta\rangle\!\rangle \bigcirc \top$ to Δ;
(3) for each state Δ obtained at steps 1 and 2, put $\Gamma \Longrightarrow \Delta$;
(4) if, however, the pretableau already contains a state Δ' that coincides with Δ, do not create another copy of Δ', but only put $\Gamma \Longrightarrow \Delta'$.

We denote the finite set of states that have outgoing edges from a prestate Γ by **states**(Γ). These include genuinely "new" states created by applying of **(SR)** to Γ as well as the states that had already been in the pretableau and got identified with a state that would otherwise have been created by applying **(SR)** to Γ.

Example 1. As a running example illustrating our tableau procedure, we will be constructing a tableau for the formula $\theta_1 = \neg\langle\!\langle 1\rangle\!\rangle \Box p \wedge \langle\!\langle 1,2\rangle\!\rangle \bigcirc p \wedge \neg\langle\!\langle 2\rangle\!\rangle \bigcirc \neg p$. The construction of the tableau for this formula starts off with the creation of a prestate $\Gamma_1 = \{\neg\langle\!\langle 1\rangle\!\rangle \Box p \wedge \langle\!\langle 1,2\rangle\!\rangle \bigcirc p \wedge \neg\langle\!\langle 2\rangle\!\rangle \bigcirc \neg p\}$. Next, **(SR)** is applied to Γ_1, which produces two states, which we call, for future reference, Δ_1 and Δ_2 (in the diagram below, as well as in the following examples, we omit the customary set-theoretic curly brackets around states and prestates of the (pre)tableaux):

$$(\Gamma_1) \quad \neg\langle\!\langle 1\rangle\!\rangle \Box p \wedge \langle\!\langle 1,2\rangle\!\rangle \bigcirc p \wedge \neg\langle\!\langle 2\rangle\!\rangle \bigcirc \neg p = \theta_1$$

$$\swarrow\!\!\!\swarrow \qquad\qquad \searrow\!\!\!\searrow$$

$$(\Delta_1) \quad \begin{array}{c}\theta_1, \neg\langle\!\langle 1\rangle\!\rangle \Box p, \langle\!\langle 1,2\rangle\!\rangle \bigcirc p,\\ \neg\langle\!\langle 2\rangle\!\rangle \bigcirc \neg p, \neg\langle\!\langle 1\rangle\!\rangle \bigcirc \langle\!\langle 1\rangle\!\rangle \Box p\end{array} \qquad (\Delta_2) \quad \begin{array}{c}\theta_1, \neg\langle\!\langle 1\rangle\!\rangle \Box p, \langle\!\langle 1,2\rangle\!\rangle \bigcirc p,\\ \neg\langle\!\langle 2\rangle\!\rangle \bigcirc \neg p, \neg p\end{array}$$

In general, if at least one subformula of a non-primitive member of a prestate Γ is a β-formula, Γ will have more than one minimal downward saturated extension; hence, for such a Γ, the set **states**(Γ) will contain more than one state. The only exception to this general rule may occur when we come across β-formulae for which $\beta_1 = \beta_2$, such as $(\varphi \to \neg\varphi)$.

We now turn to our second construction rule, **(Next)**, which creates "successor" prestates from states. The rule has to ensure that a sufficient supply of successor prestates is created to enforce the truth of all "next-time formulae" (see below) at the current state. Unlike the case of logics whose models are sets of states connected by edges of binary relations, such as **LTL** and **CTL**, in **ATL** successor prestates cannot be created by simply removing the "next-time" modality from a formula and creating an edge associated with that formula. On the contrary, in **ATL**, transitions are effected by move vectors, with which we, then, associate formulae made true by actions of agents making up that particular move vector. Thus, the rule **(Next)** needs to provide each agent mentioned in the input formula with a sufficient number of actions available at the current state, and then "populate" prestates associated each resultant move vector σ with appropriate formulae.

Before formally introducing the rule, we provide some intuition behind it. The rule is applicable to a state, say Δ; more precisely, it is applicable to the formulae of the form $\langle\!\langle A\rangle\!\rangle \bigcirc \varphi$—which we refer to as *positive next-time formulae*—and $\neg\langle\!\langle A\rangle\!\rangle \bigcirc \psi$, where $A \neq \Sigma$—which we refer to as *proper negative next-time formulae*—belonging to Δ. Positive and proper negative next-time formulae are referred to collectively as *next-time formulae.* These formulae are arranged in a list $\mathbb{L}$ and, thus, numbered; all the positive formulae in $\mathbb{L}$ precede all the negative ones; otherwise, the ordering is immaterial. The agents mentioned in the input formula θ can be thought of as having to decide which formulae from Δ appearing under the "next-time" coalition modalities $\langle\langle \ldots \rangle\rangle \bigcirc$ and $\neg\langle\langle \ldots \rangle\rangle \bigcirc$ should be included into a successor prestate associated with each move vector σ (inclusion into a prestate intuitively corresponds to satisfiability in the successor states of a Hintikka structure, as prestates eventually get unwound into tableau states). Therefore, the number of "actions" each agent mentioned in θ is given at Δ equals the number of the next-time formulae in Δ (= length of $\mathbb{L}$). These actions are combined into "move vectors" σ leading to successor prestates. The inclusion of formulae into the prestate Γ_σ created as a successor of Δ by an arrow labeled with σ is then decided as follows. A formula φ for which $\langle\langle A \rangle\rangle \bigcirc \varphi \in \mathbb{L}$ is included into Γ_σ, if every agent in A "votes" in σ for this formula (i.e. every ith slot in σ with $i \in A$ contains the number representing the position of $\langle\langle A \rangle\rangle \bigcirc \varphi$ in $\mathbb{L}$). On the other hand, $\neg\psi$ for which $\neg\langle\langle A \rangle\rangle \bigcirc \psi \in \mathbb{L}$ is included into Γ_σ (for technical reasons, at most one such formula can be included into any prestate) if every agent *not in* A votes, in the sense explained above for the positive case, for a negative formula from $\mathbb{L}$ (not necessarily $\neg\langle\langle A \rangle\rangle \bigcirc \psi$) and, moreover, $\neg\langle\langle A \rangle\rangle \bigcirc \psi$ is the formula decided on by the *collective* (negative) vote of agents in $\Sigma \setminus A$. Technically, this collective vote is represented by the number $\mathbf{neg}(\sigma)$, which is computed using all negative votes of σ, which allows it to represent a truly collective decision.

We now turn to the technical presentation of **(Next)**. The rule does not apply to the states containing patent inconsistencies since such states, obviously, cannot be part of any CGHS (so, we are not wasting time creating "junk" that will have to be removed anyway).

(Next) Given a state Δ such that for no χ we have $\chi, \neg\chi \in \Delta$, do the following:

(1) Order linearly all positive and proper negative next-time formulae of Δ in such a way that all the positive next-time formulae precede all the negative ones; suppose the result is the list

$$\mathbb{L} = \langle\langle A_0 \rangle\rangle \bigcirc \varphi_0, \ldots, \langle\langle A_{m-1} \rangle\rangle \bigcirc \varphi_{m-1}, \neg\langle\langle A'_0 \rangle\rangle \bigcirc \psi'_0, \ldots, \neg\langle\langle A'_{l-1} \rangle\rangle \bigcirc \psi_{l-1}.$$

(Note that, due to step 2 of **(SR)**, $\mathbb{L}$ is always non-empty.) Let $r_\Delta = m + l$; denote by $D(\Delta)$ the set $\{0, \ldots, r_\Delta - 1\}^{|\Sigma_\theta|}$; lastly, for every $\sigma \in D(\Delta)$, denote by $N(\sigma)$ the set $\{\, i \mid \sigma_i \geq m \,\}$, where σ_i stands for the ith component of the tuple σ, and by $\mathbf{neg}(\sigma)$ the number $[\sum_{i \in N(\sigma)} (\sigma_i - m)] \mod l$.

(2) Consider the elements of $D(\Delta)$ in the lexicographic order and for each $\sigma \in D(\Delta)$ do the following:

(a) Create a prestate

$$\begin{aligned}\Gamma_\sigma \;=\; & \{\, \varphi_p \mid \langle\langle A_p \rangle\rangle \bigcirc \varphi_p \in \Delta \text{ and } \sigma_a = p \text{ for all } a \in A_p \,\} \\ \cup\; & \{\, \neg\psi_q \mid \neg\langle\langle A'_q \rangle\rangle \bigcirc \psi_q \in \Delta,\ \mathbf{neg}(\sigma) = q, \text{ and } \Sigma_\theta - A'_q \subseteq N(\sigma) \,\};\end{aligned}$$

put $\Gamma_\sigma := \{\top\}$ if the sets on both sides of the union sign above are empty.

(b) Connect Δ to Γ_σ with $\xrightarrow{\sigma}$;

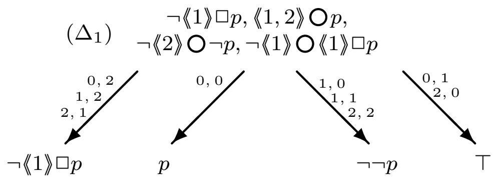
If, however, $\Gamma_\sigma = \Gamma$ for some prestate Γ that has already been added to the pretableau, only connect Δ to Γ with $\xrightarrow{\sigma}$.

We denote the finite set of prestates $\{ \Gamma \mid \Delta \overset{\sigma}{\longrightarrow} \Gamma \text{ for some } \sigma \in D(\Delta) \}$ by **prestates**(Δ). Note that a state Δ may get connected to some $\Gamma \in$ **prestates**(Δ) by arrows labeled by distinct $\sigma, \sigma' \in D(\Delta)$. In such cases, we "glue together" arrows labeled by σ and σ', in effect creating an arrow marked by a set of labels rather than a label (in examples below, in such cases, we attach several labels to a single arrow).

Example 1 continued. Let us apply the **(Next)** rule to the state $\Delta_1 = \{\theta_1, \neg\langle\langle 1\rangle\rangle\Box p, \langle\langle 1,2\rangle\rangle\bigcirc p, \neg\langle\langle 2\rangle\rangle\bigcirc\neg p, \neg\langle\langle 1\rangle\rangle\bigcirc\langle\langle 1\rangle\rangle\Box p\}$ from our running example. We arrange all the positive and proper negative next-time formulae of this state in the list $\mathbb{L} = \langle\langle 1,2\rangle\rangle\bigcirc p, \neg\langle\langle 2\rangle\rangle\bigcirc\neg p, \neg\langle\langle 1\rangle\rangle\bigcirc\langle\langle 1\rangle\rangle\Box p$. Then, at Δ_1, each of the two agents from θ_1 is going to have 3 actions, denoted by numbers 0, 1, and 2. To decide what formulae are to be included in the prestates resulting from tuples of those actions, we also need to separately number all the negative next-time formulae from $\mathbb{L}$: $\neg\langle\langle 2\rangle\rangle\bigcirc\neg p$ will be numbered 0, while $\neg\langle\langle 1\rangle\rangle\bigcirc\langle\langle 1\rangle\rangle\Box p$ will be numbered 1 (**neg**(σ) in the table below will refer to these numbers). The following table illustrates which formulae are included into prestates associated with what move vectors at Δ:

σ	**neg**(σ)	formulae
$0,0$	0	p
$0,1$	0	$\top$
$0,2$	1	$\neg\langle\langle 1\rangle\rangle\Box p$
$1,0$	0	$\neg\neg p$
$1,1$	0	$\neg\neg p$
$1,2$	1	$\neg\langle\langle 1\rangle\rangle\Box p$
$2,0$	1	$\top$
$2,1$	1	$\neg\langle\langle 1\rangle\rangle\Box p$
$2,2$	0	$\neg\neg p$

In the table above, it so happens that only one formula is included into each prestate; in general, however, this does not have to be the case. Based on the above table, by applying **(Next)** to Δ_1, we produce the following set of its prestate successors:



Remark 4.3. Technically, **(Next)** ensures that every $\Gamma_\sigma \in$ **prestates**(Δ) satisfies the following properties:

— if $\{\langle\langle A_i\rangle\rangle\bigcirc\varphi_i, \langle\langle A_j\rangle\rangle\bigcirc\varphi_j\} \subseteq \Delta$ and $\{\varphi_i, \varphi_j\} \subseteq \Gamma_\sigma$, then $A_i \cap A_j = \emptyset$;

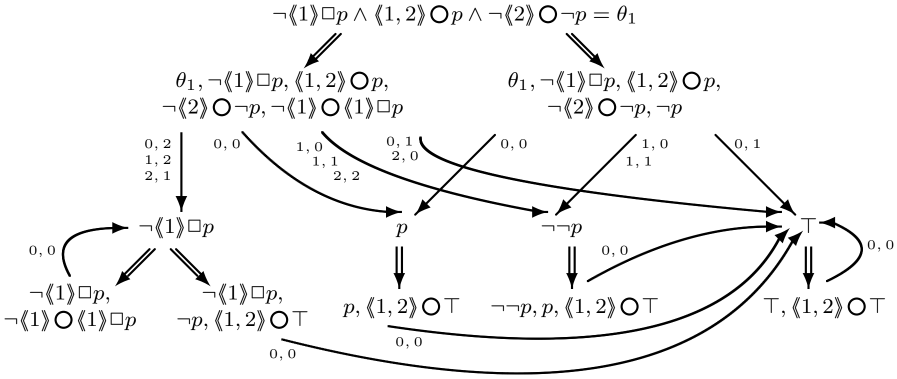
— Γ_σ contains at most one formula of the form $\neg\psi$ such that $\neg\langle\langle A\rangle\rangle\bigcirc\psi \in \Delta$, since the number **neg**(σ) is uniquely determined for every $\sigma \in D(\Delta)$;

— if $\{\langle\langle A_i\rangle\rangle\bigcirc\varphi_i, \neg\langle\langle A'\rangle\rangle\bigcirc\psi\} \subseteq \Delta$ and $\{\varphi_i, \neg\psi\} \subseteq \Gamma_\sigma$, then $A_i \subseteq A'$.

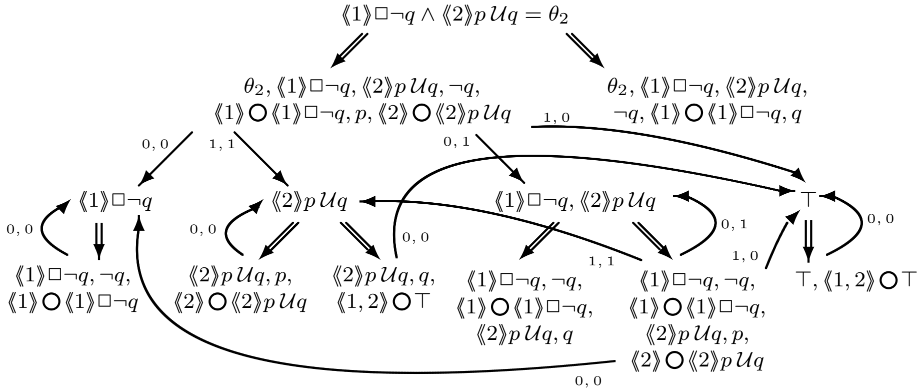
Note that there is a connection between the above properties and the basic properties of "next-time" coalition modalities, such as monotonicity and superadditivity (see [Pauly 2001a], [Pauly 2002], [Goranko and van Drimmelen 2006]).

The construction phase, starting with a single prestate $\{\theta\}$, consists of alternately applying the rule **(SR)** to the prestates created as a result of the last application of **(Next)** (or, if we are at the beginning of the whole construction, to $\{\theta\}$) and applying **(Next)** to the states created as a result of the last application of **(SR)**. This cycle continues until any application of **(Next)** does not produce any new prestates; after adding the relevant arrows, if any, the construction stage is over. As we show in the next subsection, this is bound to happen in a finite number of steps—more precisely, in the number of steps exponential in the length of θ.

Example 1 continued. Here is a complete pretableau for the formula $\theta_1 = \neg\langle\langle 1\rangle\rangle\Box p \wedge \langle\langle 1,2\rangle\rangle\bigcirc p \wedge \neg\langle\langle 2\rangle\rangle\bigcirc\neg p$:



Example 2. For yet another demonstration of our procedure, let us build a pretableau for the formula $\theta_2 = \langle\langle 1\rangle\rangle\Box\neg q \wedge \langle\langle 2\rangle\rangle p\,\mathcal{U} q$:



4.3 Termination and complexity of the construction phase

To prove that the construction phase eventually terminates and to estimate its complexity, we use the concept of the extended closure of an **ATL**-formula.

Definition 4.4. Let θ be an **ATL**-formula. The *closure of* θ, denoted by $\mathsf{cl}(\theta)$, is the least set of formulae such that

— $\theta \in \mathsf{cl}(\theta)$;

— $\mathsf{cl}(\theta)$ is closed under subformulae;
— if $\langle\!\langle A\rangle\!\rangle(\varphi\,\mathcal{U}\psi) \in \mathsf{cl}(\theta)$, then $\varphi \wedge \langle\!\langle A\rangle\!\rangle\bigcirc\langle\!\langle A\rangle\!\rangle(\varphi\,\mathcal{U}\psi) \in \mathsf{cl}(\theta)$;
— if $\neg\langle\!\langle A\rangle\!\rangle(\varphi\,\mathcal{U}\psi) \in \mathsf{cl}(\theta)$, then $\neg\psi \wedge \neg\varphi, \neg\psi \wedge \neg\langle\!\langle A\rangle\!\rangle\bigcirc\langle\!\langle A\rangle\!\rangle(\varphi\,\mathcal{U}\psi) \in \mathsf{cl}(\theta)$;
— if $\langle\!\langle A\rangle\!\rangle\Box\varphi \in \mathsf{cl}(\theta)$, then $\varphi \wedge \langle\!\langle A\rangle\!\rangle\bigcirc\langle\!\langle A\rangle\!\rangle\Box\varphi \in \mathsf{cl}(\theta)$.

Definition 4.5. Let θ be an **ATL**-formula. The *extended closure of* θ, denoted by $\mathsf{ecl}(\theta)$, is the least set of formulae such that

— if $\varphi \in \mathsf{cl}(\theta)$, then $\varphi, \neg\varphi \in \mathsf{ecl}(\theta)$;
— if $\neg\langle\!\langle\Sigma_\theta\rangle\!\rangle\bigcirc\varphi \in \mathsf{cl}(\theta)$, then $\langle\!\langle\emptyset\rangle\!\rangle\bigcirc\neg\varphi \in \mathsf{ecl}(\theta)$;
— $\top \in \mathsf{ecl}(\theta)$;
— $\langle\!\langle\Sigma\rangle\!\rangle\bigcirc\top \in \mathsf{ecl}(\theta)$.

We denote the cardinality of $\mathsf{ecl}(\theta)$ by $|\mathsf{ecl}(\theta)|$ and the length of a formula θ by $|\theta|$. When calculating the length of a formula, we assume that every agent's name counts as one symbol and that a pair of coalition braces is "lumped together" as one symbol with the temporal operator that follows it; thus, $|\langle\!\langle 1,2\rangle\!\rangle\bigcirc p| = 4$.

LEMMA 4.6. *Let θ be a **ATL**-formula. Then,* $\mathsf{ecl}(\theta)$ *is finite; more precisely,* $|\mathsf{ecl}(\theta)| \in \mathcal{O}(|\theta|)$, *i.e,* $|\mathsf{ecl}(\theta)| \leq c \cdot |\theta|$ *for some* $c \geq 1$.

PROOF. Straightforward. □

To simplify notation, let us denote $|\theta|$ by n and $|\Sigma_\theta|$ by k; let also c be the constant from the statement of the preceding lemma. While building the pretableau $\mathcal{P}^\theta$, we create $\mathcal{O}(2^{cn})$ states and $\mathcal{O}(2^{cn})$ prestates. To create a state, we need no more than $\mathcal{O}(cn)$ steps, thus the creation of all the states takes not more than $\mathcal{O}(cn \times 2^{cn})$ steps. For a given state Δ, to create all the prestates in **prestates**(Δ), we first produce a Γ_σ associated with a given $\sigma \in D(\Delta)$, which costs $\mathcal{O}(cn)$ steps, and then check whether it is identical to a prestate created earlier, which takes $\mathcal{O}((cn)^2 \times 2^{cn})$ steps. As there are, all in all, $\mathcal{O}((cn)^k)$ move vectors in $D(\Delta)$, the whole procedure of creating prestates from a given state costs $\mathcal{O}((cn)^k \times (cn + (cn)^2 \times 2^{cn}))$. Applying this procedure to all $\mathcal{O}(2^{cn})$ states, i.e, creating all prestates can thus be done in $\mathcal{O}(2^{cn} \times (cn)^k \times (cn + (cn)^2 \times 2^{cn}) = \mathcal{O}(2^{(k+1)\log(cn)+cn} + 2^{(k+2)\log(cn)+2cn}) = \mathcal{O}(2^{(k+2)\log(cn)+2cn})$. As this clearly dominates the complexity of creating states, the cost of the construction phase as a whole is $\mathcal{O}(2^{(k+2)\log(cn)+2cn})$.

4.4 Prestate elimination phase

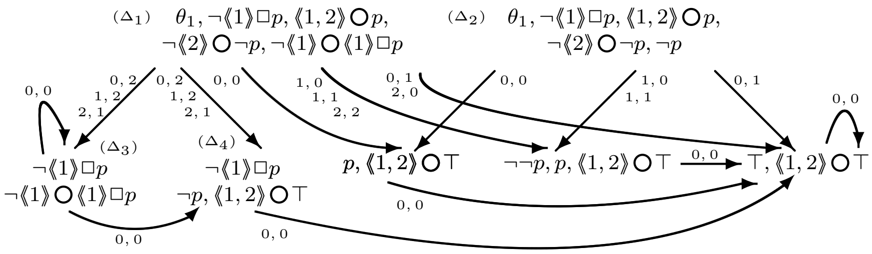
At the second phase of the tableau procedure, we remove from $\mathcal{P}^\theta$ all the prestates and all the unmarked arrows, by applying the following rule:

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

(1) remove Γ from $\mathcal{P}^\theta$;
(2) for all states Δ in $\mathcal{P}^\theta$ with $\Delta \overset{\sigma}{\longrightarrow} \Gamma$ and all $\Delta' \in$ **states**(Γ), put $\Delta \overset{\sigma}{\longrightarrow} \Delta'$.

We call the graph obtained by applying **(PR)** to $\mathcal{P}^\theta$ the *initial tableau*, which we denote by $\mathcal{T}_0^\theta$. Note that if in $\mathcal{P}^\theta$ we have $\Delta \overset{\sigma}{\longrightarrow} \Gamma$ and **states**(Γ) contains more than one state, then in $\mathcal{T}_0^\theta$ there is going to be more than one edge labeled with σ going out of Δ.

Example 1 continued. Here is the initial tableau $\mathcal{T}_0^{\theta_1}$ for the formula $\theta_1 = \neg\langle\!\langle 1\rangle\!\rangle\Box p \wedge \langle\!\langle 1,2\rangle\!\rangle\bigcirc p \wedge \neg\langle\!\langle 2\rangle\!\rangle\bigcirc\neg p$ (as before, some states are named for future reference):



Thus, our procedure for the formula $\neg\langle\!\langle 1\rangle\!\rangle\Box p \wedge \langle\!\langle 1,2\rangle\!\rangle\bigcirc p \wedge \neg\langle\!\langle 2\rangle\!\rangle\bigcirc\neg p$ creates 7 states. For the sake of comparison with the decision procedure from [Walther et al. 2006], which as already mentioned, follows the top-down tableaux approach, we estimate how many states would be created using that procedure. As the running time of both procedures is roughly proportional to the number of states created, this should give us an idea as to how the two procedures compare in practice.

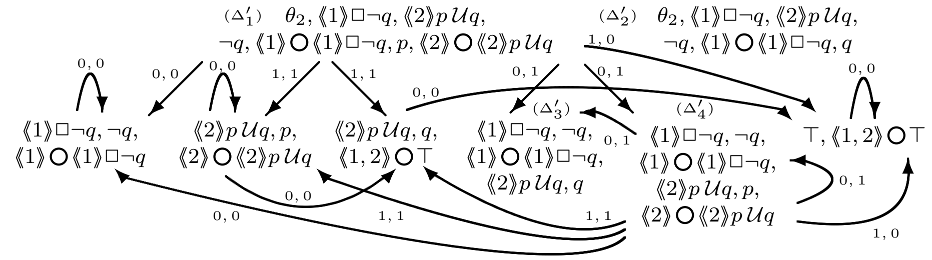
While we use the concept of extended closure of a formula for metatheoretical purposes (to prove termination and estimate complexity, see Section 4.3), the top-down tableaux-like decision procedure from [Walther et al. 2006] uses the extended closure of a formula θ to actually run the procedure for θ. Technically speaking, the procedure from [Walther et al. 2006] creates not states, but "types"—maximal, propositionally consistent, saturated subsets of the extended closure of the input formula. So, we estimate how many types the procedure from [Walther et al. 2006] would create for the formula $\neg\langle\!\langle 1\rangle\!\rangle\Box p \wedge \langle\!\langle 1,2\rangle\!\rangle\bigcirc p \wedge \neg\langle\!\langle 2\rangle\!\rangle\bigcirc\neg p$. To that end, we fist enumerate positive formulas of the extended closure for this formula (the definition of extended closure in [Walther et al. 2006] is slightly different from ours):

(1) $\neg\langle\!\langle 1\rangle\!\rangle\Box p \wedge \langle\!\langle 1,2\rangle\!\rangle\bigcirc p \wedge \neg\langle\!\langle 2\rangle\!\rangle\bigcirc p$,
(2) $\neg\langle\!\langle 1\rangle\!\rangle\Box p \wedge \langle\!\langle 1,2\rangle\!\rangle\bigcirc p$,
(3) $\langle\!\langle 2\rangle\!\rangle\bigcirc\neg p$,
(4) $\langle\!\langle 1\rangle\!\rangle\Box p$,
(5) $\langle\!\langle 1\rangle\!\rangle\bigcirc\langle\!\langle 1\rangle\!\rangle\Box p$,
(6) $\langle\!\langle 1,2\rangle\!\rangle\bigcirc p$,
(7) p.

For every formula from the above list, each type contains either that formula or its negation. However, not every such combination is allowed, as there are dependencies between formulae as to their presence in a type.

First, if (1) is in a type, then that type must contain (2), $\neg$(3), $\neg$(4) and (6); so, there are 2^2 distinct types containing formula (1). Second, if $\neg$(1) and $\neg$(2) are in a type, then we have two cases: if the type contains (4), then it contains (5), generating 2^3 types; and if the type $\neg$(6), then it contains $\neg$(4), generating 2^3 more types. Lastly, if $\neg$(1) and (2) are in a type, then (3), $\neg$(4), and (6) are also in the type, generating 2^2 types. Thus, all in all, the top-down tableau procedure from [Walther et al. 2006] creates 24 types, as opposed to 7 states created by incremental tableaux.

Example 2 continued. Here is the initial tableau $\mathcal{T}_0^{\theta_2}$ for the formula $\theta_2 = \langle\langle 1\rangle\rangle\Box\neg q \wedge \langle\langle 2\rangle\rangle p\,\mathcal{U} q$ (as in the previous example, some states are named for future reference):



Again, for the sake of comparison with top-down tableaux form [Walther et al. 2006], we estimate the number of types created by that procedure; a calculation similar to the one from the previous example shows that 36 types are created by the top-down tableaux, as opposed to 8 states created by the incremental tableaux[8].

We briefly remark on the time required for this second phase. Once again, to simplify notation, let us denote $|\theta|$ by n. Recall that $|\mathsf{ecl}(\theta)| \in \mathcal{O}(|\theta|)$, i.e, $|\mathsf{ecl}(\theta)| = c\cdot|\theta|$ for some $c \geq 1$. To remove a single prestate, we need to delete from the memory its $\mathcal{O}(cn)$ formulae and redirect at most $\mathcal{O}(2^{cn} \times 2^{cn})$ edges—having identified set-theoretically equal states as part of the application of **(Next)** and having "glued together" arrows having the same source and target, we do not have, at this stage, to deal with $\mathcal{O}(cn^k)$ outgoing edges for each state. Hence, the removal of a single prestate can be done in $\mathcal{O}(2^{2cn})$ steps. As there are at most $\mathcal{O}(2^{cn})$ prestates, the whole procedure takes $\mathcal{O}(2^{3cn})$ steps.

4.5 State elimination phase

During the state elimination phase, we remove those nodes of $\mathcal{T}_0^\theta$ that cannot be satisfied in any CGHS. As already mentioned, there are three reasons why a state Δ of $\mathcal{T}_0^\theta$ can turn out to be unsatisfiable in any CGHS. First, Δ may contain a patent inconsistency[9]. Secondly, satisfiability of Δ may require that at least one state from a set of tableau states X is satisfiable as a successor of the state s_Δ of a CGHS presumably satisfying Δ, while all states of X turn out to be unsatisfiable sets. Thirdly, Δ may contain an eventuality that is not realized in the tableau; that this implies unsatisfiability of Δ is much less obvious than in the preceding two cases—in fact, a major task within the soundness proof for our procedure is to establish that this is indeed so. Accordingly, we have three elimination rules, **(E1)**–**(E3)**, each taking care of eliminating states of $\mathcal{T}_0^\theta$ on one of the above-mentioned counts.

[8] Clearly, our argument that incremental tableaux, on average, are more efficient than the top-down ones relies on "common sense" and does not constitute a mathematical proof. No such proof can be given in abstract, as any average-case analysis would require a fixed, presumably application-related, probability distribution defined on **ATL**-formulae.

[9] As states are downward-saturated, this is tantamount to saying that Δ contains a propositional inconsistency, even though in general these two concepts are not identical, as noted earlier.

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

The rationale for the first rule is obvious.

(E1) If $\{\varphi, \neg\varphi\} \subseteq \Delta \in S_n^\theta$, then obtain $\mathcal{T}_{n+1}^\theta$ by eliminating Δ from $\mathcal{T}_n^\theta$.

The rationale behind the second rule is also intuitively clear: if Δ is to be satisfiable, then for each $\sigma \in D(\Delta)$ there should exists a satisfiable Δ' with $\Delta \xrightarrow{\sigma} \Delta'$. If all such Δ's have been eliminated because they are unsatisfiable, then Δ is itself unsatisfiable.

(E2) If, for some $\sigma \in D(\Delta)$, all states Δ' with $\Delta \xrightarrow{\sigma} \Delta'$ have been eliminated at earlier stages, then obtain $\mathcal{T}_{n+1}^\theta$ by eliminating Δ from $\mathcal{T}_n^\theta$.

To formulate **(E3)**, we need the concept of realization of an eventuality in a tableau. To define that concept, we need some auxiliary notation. Let $\Delta \in S_0^\theta$, and let $\langle\langle A\rangle\rangle\bigcirc\varphi$ be the p-th formula in the linear ordering of the next-time formulae of Δ induced as part of application of **(Next)** to Δ; let, finally, $\neg\langle\langle A'\rangle\rangle\bigcirc\psi$ be the q-th formula in the same ordering. Then, we use the following notation:

$$D(\Delta, \langle\langle A\rangle\rangle\bigcirc\varphi) := \{\, \sigma \in D(\Delta) \mid \sigma_a = p \text{ for every } a \in A \,\};$$
$$D(\Delta, \neg\langle\langle A'\rangle\rangle\bigcirc\psi) := \{\, \sigma \in D(\Delta) \mid \mathbf{neg}(\sigma) = q \text{ and } \Sigma_\theta \setminus A' \subseteq N(\sigma) \,\}.$$

Intuitively, $D(\Delta, \chi)$ corresponds to an A-move (if $\chi = \langle\langle A\rangle\rangle\bigcirc\varphi$) or a co-$A$-move (if $\chi = \neg\langle\langle A'\rangle\rangle\bigcirc\psi$) witnessing the "satisfaction" of χ at state Δ (recall that A-moves and co-A-moves can be identified with equivalence classes on the set of move vectors).

We now recursively define what it means for an eventuality of the form $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$ to be realized at a state Δ of tableau $\mathcal{T}_n^\theta$.

Definition 4.7 *Realization of eventuality* $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$.

(1) If $\{\psi, \langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi\} \subseteq \Delta \in S_n^\theta$, then $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$ is *realized* at Δ in $\mathcal{T}_n^\theta$;
(2) If $\{\varphi, \langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi, \langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi\} \subseteq \Delta$ and for every $\sigma \in D(\Delta, \langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi)$, there exists $\Delta' \in S_n^\theta$ such that
— $\Delta \xrightarrow{\sigma} \Delta'$ and
— $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$ is realized at Δ' in $\mathcal{T}_n^\theta$,
then $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$ is *realized* at Δ in $\mathcal{T}_n^\theta$.

The definition of realization for eventualities of the form $\neg\langle\langle A\rangle\rangle\Box\varphi$ is analogous:

Definition 4.8 *Realization of eventuality* $\neg\langle\langle A\rangle\rangle\Box\varphi$.

(1) If $\{\neg\varphi, \neg\langle\langle A\rangle\rangle\Box\varphi\} \subseteq \Delta \in S_n^\theta$, then $\neg\langle\langle A\rangle\rangle\Box\varphi$ is *realized* at Δ in $\mathcal{T}_n^\theta$;
(2) If $\{\neg\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\Box\varphi, \neg\langle\langle A\rangle\rangle\Box\varphi\} \subseteq \Delta$ and, for every $\sigma \in D(\Delta, \neg\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\Box\varphi)$ there exists $\Delta' \in S_n^\theta$ such that
— $\Delta \xrightarrow{\sigma} \Delta'$ and
— $\neg\langle\langle A\rangle\rangle\Box\varphi$ is realized at Δ' in $\mathcal{T}_n^\theta$,

then $\neg\langle\langle A\rangle\rangle\Box\varphi$ is *realized* at Δ in $\mathcal{T}_n^\theta$.

We can now state our third elimination rule.

(E3) If $\Delta \in S_n^\theta$ contains an eventuality that is not realized at Δ in $\mathcal{T}_n^\theta$, then obtain $\mathcal{T}_{n+1}^\theta$ by removing Δ from $\mathcal{T}_n^\theta$.

While implementation of the rules **(E1)** and **(E2)** is straightforward, implementation of **(E3)** is less so. It can be done by computing the set of states realizing a given eventuality ξ in tableau $\mathcal{T}_n^\theta$, say, by marking those states that realize ξ in $\mathcal{T}_n^\theta$. To formally describe the procedure, we need some extra notation.

First, given $\Delta \in S_n^\theta$ and $\sigma \in D(\Delta)$, we denote by $\mathbf{succ}_\sigma(\Delta)$ the set $\{\, \Delta' \in S_n^\theta \mid \Delta \xrightarrow{\sigma} \Delta' \,\}$. Secondly, given a formula χ, we write, abusing set-theoretic notation, $\chi \in \mathcal{T}_n^\theta$ to mean that $\chi \in \Delta$ for some $\Delta \in S_n^\theta$.

We now describe the marking procedure for $\mathcal{T}_n^\theta$ with respect to eventuality ξ. We first do so for eventualities of the form $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$. Initially, we mark Δ if $\psi \in \Delta$. Afterwards, we repeat the following computation for every $\Delta \in S_n^\theta$ that is still unmarked: mark Δ if, for every $\sigma \in D(\Delta, \langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi)$, there exists at least one Δ' such that $\Delta' \in \mathbf{succ}_\sigma(\Delta)$ and Δ' is marked. The procedure is over when no more states can get marked.

The procedure for computing eventualities of the form $\neg\langle\langle A\rangle\rangle\Box\varphi$ is similar. Initially, we mark Δ if $\neg\varphi \in \Delta$. Afterwards, we repeat the following computation for every $\Delta \in S_n^\theta$ that is still unmarked: mark Δ if, for every $\sigma \in D(\Delta, \neg\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\Box\varphi\}$, there exists at least one Δ' such that $\Delta' \in \mathbf{succ}_\sigma(\Delta)$ and Δ' is marked. The procedure is over when no more states can get marked.

LEMMA 4.9. *Let $\Delta \in S_n^\theta$ and $\xi \in \mathcal{T}_n^\theta$ be an eventuality. Then, ξ is realized at Δ in $\mathcal{T}_n^\theta$ iff Δ is marked in $\mathcal{T}_n^\theta$ with respect to ξ.*

PROOF. Straightforward. □

Thus, the application of **(E3)** in tableau $\mathcal{T}_n^\theta$ with respect to eventuality ξ consists of carrying out the marking procedure with respect to ξ and then removing all the states that contain ξ, but have not been marked with respect to ξ.

We have thus far described individual rules and how they can be implemented. To describe the state elimination phase as a whole, it is crucial to specify the order of application of those rules.

First, we apply **(E1)** to all the states of $\mathcal{T}_0^\theta$; it is clear that, once it is done, we do not need to go back to **(E1)** again. The cases of **(E2)** and **(E3)** are slightly more involved. Having applied **(E3)** to the states of the tableau, we could have removed, for some Δ, all the states accessible from it along the arrows marked by some $\sigma \in D(\Delta)$; hence, we need to reapply **(E2)** to the resultant tableau to get rid of such Δ's. Conversely, having applied **(E2)**, we could have removed some states that were instrumental in realizing certain eventualities; hence, having applied **(E2)**, we need to reapply **(E3)**. Furthermore, we cannot stop the procedure unless we have checked that *all* eventualities are realized. Thus, what we need is to apply **(E3)** and **(E2)** in a dovetailed sequence that cycles through all the eventualities. More precisely, we arrange all the eventualities occurring in the tableau obtained from $\mathcal{T}_0^\theta$ by having applied **(E1)** to $\mathcal{T}_0^\theta$ in the list $\xi_1, \ldots, \xi_m$. Then, we proceed in cycles.

Each cycle consists of alternatingly applying **(E3)** to the pending eventuality, and then applying **(E2)** to the tableau resulting from that application, until all the eventualities have been dealt with; once we reach ξ_m, we loop back to ξ_1. The cycles are repeated until, having gone through the whole cycle, we have not had to remove any states.

Once that happens, the state elimination phase is over. The resultant graph we call the *final tableau* for θ and denote by $\mathcal{T}^\theta$.

Definition 4.10. The final tableau $\mathcal{T}^\theta$ is *open* if $\theta \in \Delta$ for some $\Delta \in S^\theta$; otherwise, $\mathcal{T}^\theta$ is *closed*.

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

Example 1 continued. Consider the initial tableau for our formula θ_1. First, no states of that tableau contain patent inconsistencies. Moreover, all four states containing the eventuality $\neg\langle\!\langle 1 \rangle\!\rangle\Box p$ (which is the only eventuality in the tableau) get marked with respect to $\neg\langle\!\langle 1 \rangle\!\rangle\Box p$. Indeed, Δ_2 and Δ_4 get marked since they contain $\neg p$; Δ_1 get marked since all the relevant move vectors (i.e, those for which $\mathbf{neg}(\sigma) = 1$ and agent 2 votes negatively; there are 3 such move vectors: $(0, 2), (1, 2), (2, 1)$) lead to a state Δ_4 that is marked; finally Δ_3 is marked as the only relevant move vector connects it to a marked state, Δ_4. Lastly, all the states have all the required successors. Therefore, no state of the initial tableau gets eliminated, hence, the final tableau $\mathcal{T}^{\theta_1}$ coincides with the initial tableau $\mathcal{T}_0^{\theta_1}$. Thus, $\mathcal{T}^{\theta_1}$ is open (it contains two states, Δ_1 and Δ_2, containing θ_1); therefore, $\theta_1 = \neg\langle\!\langle 1 \rangle\!\rangle\Box p \wedge \langle\!\langle 1, 2 \rangle\!\rangle \bigcirc p \wedge \neg\langle\!\langle 2 \rangle\!\rangle \bigcirc \neg p$ is satisfiable.

Example 2 continued. Consider the initial tableau for the formula θ_2. We have to eliminate state Δ'_2 due to **(E1)**, as it contains a patent inconsistency. For the same reason, we have to eliminate Δ'_3. Furthermore, state Δ'_4 gets eliminated due to **(E3)** since it contains an eventuality $\langle\!\langle 2 \rangle\!\rangle p\,\mathcal{U} q$, but does not get marked with respect to it, as the only consistent state reachable from Δ'_4 along the "relevant" move vector $(0, 1)$, which is Δ'_4 itself, does not contain q. Then, Δ'_1 has to be eliminated, as all states reachable form it along the move vector $(0, 1)$ have been eliminated. Thus, all the states containing the input formula, namely Δ'_1 and Δ'_2, are eliminated from the tableau. Therefore, the final tableau for θ_2 is closed and, hence, $\theta_2 = \langle\!\langle 1 \rangle\!\rangle\Box\neg q \wedge \langle\!\langle 2 \rangle\!\rangle p\,\mathcal{U} q$ is unsatisfiable.

4.6 Incremental tableaux for CTL

The branching-time logic **CTL** can be regarded as the one-agent version of **ATL**, where $\langle\!\langle \emptyset \rangle\!\rangle$ is the universal path quantifier and $\langle\!\langle 1 \rangle\!\rangle$ is the existential path quantifier. Thus, after due simplifications (notably, of the rule **(Next)**), our tableau method produces an incremental tableau procedure for **CTL**, which is practically more efficient (in the average case) than Emerson and Halpern's top-down tableau from [Emerson and Halpern 1985].

4.7 Complexity of the procedure

We now estimate the complexity of the tableau procedure described above. As before, let $n = |\theta|$, $k = |\Sigma_\theta|$, and let c be the constant from the equation $|\mathsf{ecl}(\theta)| = c \cdot |\theta|$ (recall Lemma 4.6).

As we have seen, the costs of the construction phase and of the prestate elimination phase are, respectively, $\mathcal{O}(2^{(k+2)\log(cn)+2cn})$ and $\mathcal{O}(2^{3cn})$ steps. It, thus, remains to estimate the time required for the state elimination phase. During that phase, we first apply **(E1)** to every state of the initial tableau. To do that, we need to go through $\mathcal{O}(2^{cn})$ states, and for each formula φ of each state Δ check whether $\neg\varphi \in \Delta$; this can be done in time $\mathcal{O}(2^{cn} \times (cn)^2) = \mathcal{O}(2^{2\log(cn)+cn})$.

Next, we embark on the sequence of dovetailed applications of **(E3)** and **(E2)**. We do it in cycles, whose number is bounded by $\mathcal{O}(2^{cn})$, each cycle involving going through all the eventualities, whose number is bounded by $\mathcal{O}(cn)$. For each eventuality ξ, we have to, first, run the marking procedure with respect to ξ and then remove, as prescribed by **(E3)**, all the relevant unmarked states; then, we apply the procedure implementing **(E2)**. The latter procedure can be carried out in $\mathcal{O}(2^{cn} \times (cn^k + cn)) = \mathcal{O}(2^{k\log(cn)+n} + 2^{\log(cn)+cn}) = \mathcal{O}(2^{k\log(cn)+n})$ steps, as we should go through $\mathcal{O}(2^{cn})$ states, doing the check for $\mathcal{O}((cn)^k)$ moves marking outgoing arrows, and possibly deleting $\mathcal{O}(cn)$ formulas of the state. Since $k \leq n$, the cost of applying **(E2)** is bounded by $\mathcal{O}(2^{n\log(cn)+n}) = \mathcal{O}(2^{n(\log(cn)+1)})$ steps. As for the former, we need to compute the set of states realizing ξ in $\mathcal{T}^\theta_n$, which can be done in $\mathcal{O}(2^{k\log(cn)+3cn})$ steps, as we do at most $\mathcal{O}(2^{cn})$ "global" status ("marked"/ "unmarked") updates, each time updating the status for at most $\mathcal{O}(2^{cn})$ states, each of these updates requiring looking at $\mathcal{O}((cn)^k)$ possible moves, which as several outgoing arrows can be marked with the same move, can be repeated at most $\mathcal{O}(2^{cn})$ times. (For simplicity, we disregard the cost of deleting states with unrealized eventualities, as its complexity, $\mathcal{O}(2^{cn} \times cn)$, is clearly dominated by the complexity of the marking procedure.) Thus, the whole sequence of dovetailed applications of **(E2)** and **(E3)** requires $\mathcal{O}((2^{cn} \times cn) \times (2^{k\log(cn)+n} + 2^{k\log(cn)+3cn})) = \mathcal{O}(2^{(k+1)\log(cn)+4cn})$.

Thus, the overall complexity of our tableau procedure is $\mathcal{O}(2^{(k+2)\log(cn)+2cn}) + \mathcal{O}(2^{3cn}) + \mathcal{O}(2^{(k+1)\log(cn)+4cn})$. As $k \leq n+1$, this expression is bounded by $\mathcal{O}(2^{n\log n+5cn}) = \mathcal{O}(2^{2n\log n}) = \mathcal{O}(2^{2|\theta|\log|\theta|})$. This upper bound appears to be better than the one claimed in [Walther et al. 2006] for the top-down tableaux developed therein (namely, $\mathcal{O}(2^{n^2})$); a more careful analysis reveals, however, that the upper bound for tableaux from [Walther et al. 2006] is within $\mathcal{O}(2^{2n\log n})$, too.

5. SOUNDNESS AND COMPLETENESS

We now prove that the tableau procedure described above is sound and complete with respect to **ATL** semantics as defined in section 2.2; in algorithmic terminology, we show that the procedure is correct.

5.1 Soundness

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

Before going into the technical details, we give an informal outline of the proof.

The tableau procedure for the input formula θ starts off with creating a single prestate $\{\theta\}$. Then, we unwind $\{\theta\}$ into states, each of which contains θ. To establish soundness, it suffices to show that at least one 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 Γ is satisfiable, then at least one state created from Γ using **(SR)** is also satisfiable. In particular, it ensures that if θ is satisfiable, then so is at least one state obtained by **(SR)** form $\{\theta\}$. To ensure soundness, it is enough to show that this state never gets eliminated from the tableau.

To that end, we first show (Lemma 5.2) that, given a satisfiable state Δ, all the prestates created from Δ by **(Next)**—each of which is associated with a move vector, say σ—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 Δ is satisfiable, then for every move vector σ at Δ, in the initial tableau, Δ will have at least one satisfiable successor reachable by an arrow marked with σ; hence, if Δ is satisfiable, it will not be eliminated on account of **(E2)**. Lastly, we show that no satisfiable states contain unrealized eventualities (in the sense of Definitions 4.7 and 4.8), and thus cannot be removed from the tableau on account of **(E3)**. 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 θ, and thus containing θ, survives to the end of the procedure—hence, the final tableau for θ is open, as desired.

We start with the lemma that essentially asserts that the "state-creation" component of our tableaux preserves satisfiability.

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

PROOF. Straightforward (see a remark at the end of section 3.1, though). □

The next lemma shows that **(Next)** creates from satisfiable states satisfiable prestates (to see this, compare the condition of the lemma with Remark 4.3).

LEMMA 5.2. *Let $\Phi = \{\langle\langle A_1\rangle\rangle\bigcirc\varphi_1, \ldots, \langle\langle A_m\rangle\rangle\bigcirc\varphi_m, \neg\langle\langle A'\rangle\rangle\bigcirc\psi\}$ be a set of formulae such that $A_i \cap A_j = \emptyset$ for every $1 \leq i, j \leq m$ and $A_i \subseteq A'$ for every $1 \leq i \leq m$. Let $\mathcal{M}, s \Vdash \Phi$ for some CGM $\mathcal{M}$ and $s \in \mathcal{M}$. Let, furthermore, $\sigma_{A_i} \in D_{A_i}(s)$ be an A_i-move witnessing the truth of $\langle\langle A_i\rangle\rangle\bigcirc\varphi_i$ at s, for each $1 \leq i \leq m$, and let, finally, $\sigma^c_{A'} \in D^c_{A'}(s)$ be a co-A'-move witnessing the truth of $\neg\langle\langle A'\rangle\rangle\bigcirc\psi$ at s. Then, there exists $s' \in out(s, \sigma_{A_1}) \cap \ldots \cap out(s, \sigma_{A_m}) \cap out(s, \sigma^c_{A'})$ such that $\mathcal{M}, s' \Vdash \{\varphi_1, \ldots, \varphi_m, \neg\psi\}$.*

PROOF. As $A_i \cap A_j = \emptyset$ for every $1 \leq i, j \leq m$, all the moves σ_{A_i}, where $1 \leq i \leq m$, can be "fused" into a move $\sigma_{A_1 \cup \ldots \cup A_m}$. Then, as $A_i \subseteq A'$ for every $1 \leq i \leq m$, the application of the co-move $\sigma^c_{A'}$ to any extension of $\sigma_{A_1 \cup \ldots \cup A_m}$ to a move of the coalition $\Sigma_\theta \setminus A' \supseteq A_1 \cup \ldots \cup A_m$ produces a move vector σ such that $s' = \delta(s, \sigma)$ satisfies both properties from the statement of the lemma. □

The preceding two lemmas show that from satisfiable (pre)states we produce satisfiable (pre)states. This, in particular, implies two things: first, at least one of the states containing the input formula θ is satisfiable and, second, satisfiable states never get eliminated due to **(E2)**. It is also clear that a satisfiable state cannot contain a propositional inconsistency and, thus, cannot be removed due to **(E1)**.

Therefore, all that remains to show to establish soundness is that **(E3)** does not eliminate from tableaux satisfiable states. To that end, we will need some extra definitions and pieces of notation, which draw analogies between what happens in CGMs and tableaux (Definition 5.3 through Notational convention 5.5).

In what follows, we treat labels of the arrows of the tableaux as move vectors; the concepts of A-move, and all the concomitant definitions and notation are then used in exactly the same way as for CGFs (see section 2.2.1); analogously for co-A-moves (see section 2.5). We only explicitly mention what notion (i.e., the one relating to the semantics of **ATL** or to tableaux) is referred to if the context leaves room for ambiguity. The only notion that differs between **ATL**-semantics and the **ATL**-tableaux is that of "outcome" of (CGF vs. tableau) moves and co-moves. Unlike the former, the latter are generally not unique, as there might be several outgoing arrows from a state Δ labeled with the same "move vector" σ. We, however, define an outcome set of a tableau A-move σ_A to contain exactly one state obtained from Δ by following a given $\sigma \sqsupseteq \sigma_A$ to make them resemble outcomes of A-moves in CGFs.

Definition 5.3. Let $\Delta \in S_n^\theta$ and $\sigma_A \in D_A(\Delta)$. An *outcome set* of σ_A at Δ is a minimal set of states $X \subseteq S_n^\theta$ such that, for every $\sigma \sqsupseteq \sigma_A$, there exists exactly one $\Delta' \in X$ such that $\Delta \overset{\sigma}{\longrightarrow} \Delta'$.

Outcome sets for tableau co-moves are defined analogously.

Definition 5.4. Let $\Delta \in S_n^\theta$ and $\sigma_A^c \in D_A^c(\Delta)$. An *outcome set* of σ_A^c at Δ is a minimal set of states $X \subseteq S_n^\theta$ such that, for every $\sigma_A \in D_A(\Delta)$, there exists exactly one $\Delta' \in X$ such that $\Delta \overset{\sigma_A^c(\sigma_A)}{\longrightarrow} \Delta'$.

Notational convention 5.5.

(1) Whenever we write $\langle\langle A_p \rangle\rangle \bigcirc \varphi_p \in \Delta \in S_n^\theta$, we mean that $\langle\langle A_p \rangle\rangle \bigcirc \varphi_p$ is the p-th formula in the linear ordering of the next-time formulae of Δ induced as part of applying the **(Next)** rule to Δ. We use the notation $\neg\langle\langle A'_q \rangle\rangle \bigcirc \psi_q \in \Delta \in S_n^\theta$ in an analogous way.
(2) Given $\langle\langle A_p \rangle\rangle \bigcirc \varphi_p \in \Delta \in S_n^\theta$, by $\sigma_{A_p}[\langle\langle A_p \rangle\rangle \bigcirc \varphi_p]$ we denote (the unique) tableau A_p-move $\sigma_{A_p} \in D_{A_p}(\Delta)$ such that $\sigma_{A_p}(a) = p$ for every $a \in A_p$.
(3) Given a proper $\neg\langle\langle A'_q \rangle\rangle \bigcirc \psi_q \in \Delta \in S_n^\theta$, by $\sigma^c_{A'_q}[\neg\langle\langle A'_q \rangle\rangle \bigcirc \psi_q]$ we denote (the unique) tableau co-A'_q-move satisfying the following condition: $\mathbf{neg}(\sigma^c_{A'_q}(\sigma_{A'_q})) = q$ and $\Sigma_\theta \setminus A'_q \subseteq N(\sigma^c_{A'_q}(\sigma_{A'_q}))$ for every $\sigma_{A'_q} \in D_{A'_q}(\Delta)$.

We now get down to proving that **(E3)** does not eliminate any satisfiable states. We need to show that if a tableau $\mathcal{T}_n^\theta$ contains a state Δ that is satisfiable and contains an eventuality ξ, then ξ is realized at Δ. This will be accomplished by showing that $\mathcal{T}_n^\theta$ "contains" a structure (more precisely, a tree) that, in a sense to be made precise, "witnesses" the realization of ξ at Δ in $\mathcal{T}_n^\theta$. This tree will, in a sense to be made precise, emulate a tree of runs effected by a strategy or co-strategy that "realizes" an eventuality in a model. This simulation is going to be carried out step-by-step, each step, i.e. A-move (in the case of $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$) or co-$A$-move (in the case of $\neg\langle\langle A\rangle\rangle\Box\varphi$) will be simulated by a tableau move or co-move associated with a respective eventuality. That this step-by-step simulation can be done is proved in the next two lemmas (together with their corollaries).

LEMMA 5.6. *Let $\langle\langle A_p\rangle\rangle\bigcirc\varphi_p \in \Delta \in S_n^\theta$ and let $\mathcal{M}, s \Vdash \Delta$ for some CGM $\mathcal{M}$ and state $s \in \mathcal{M}$. Let, furthermore, $\sigma_{A_p} \in D_{A_p}(s)$ be an A_p-move witnessing the truth of $\langle\langle A_p\rangle\rangle\bigcirc\varphi_p$ at s. Then, there exists in $\mathcal{T}_n^\theta$ an outcome set X of $\sigma_{A_p}[\langle\langle A_p\rangle\rangle\bigcirc\varphi_p]$ such that for each $\Delta' \in X$ there exists $s' \in out(s, \sigma_{A_p})$ such that $\mathcal{M}, s' \Vdash \Delta'$.*

PROOF. Consider the set of prestates $Y = \{\,\Gamma \in \textbf{prestates}(\Delta) \mid \Delta \xrightarrow{\sigma} \Gamma$ for some $\sigma \sqsupseteq \sigma_{A_p}[\langle\langle A_p\rangle\rangle\bigcirc\varphi_p]\,\}$. Take an arbitrary $\Gamma \in Y$. It follows immediately from the **(Next)** rule (see Remark 4.3) that Γ (which must contain φ_p) is either of the form $\{\varphi_1, \ldots, \varphi_m, \neg\psi\}$, where

$$\{\langle\langle A_1\rangle\rangle\bigcirc\varphi_1, \ldots, \langle\langle A_m\rangle\rangle\bigcirc\varphi_m, \neg\langle\langle A'\rangle\rangle\bigcirc\psi\} \subseteq \Delta$$

satisfies the condition of Lemma 5.2, or of the form $\{\varphi_1, \ldots, \varphi_m\}$, where

$$\{\langle\langle A_1\rangle\rangle\bigcirc\varphi_1, \ldots, \langle\langle A_m\rangle\rangle\bigcirc\varphi_m\} \subseteq \Delta$$

and $A_i \cap A_j = \emptyset$ for every $1 \le i, j \le m$.

As $\mathcal{M}, s \Vdash \Delta$, in the former case, by Lemma 5.2, there exists $s' \in out(s, \sigma_{A_p})$ with $\mathcal{M}, s' \Vdash \Gamma$. Then Γ can be extended to a downward saturated set Δ' containing at least one next-time formula ($\langle\langle\Sigma_\theta\rangle\rangle\bigcirc\top$ if nothing else) such that $\mathcal{M}, s' \Vdash \Delta'$. This is done by choosing, for every β-formula to be dealt with, the "disjunct" that is actually true in $\mathcal{M}$ at s' (if both "disjuncts" happen to be true at s', the choice is arbitrary).

In the latter case, the same conclusion follows from Lemma 5.2 again, by adding to Δ the valid formula $\neg\langle\langle\Sigma_\theta\rangle\rangle\bigcirc\bot$.

To complete the proof of the lemma, take X to be the set of all tableau states Δ' obtainable from the prestates in Y in the way described above. □

COROLLARY 5.7. *Let $\langle\langle A_p\rangle\rangle\bigcirc\varphi_p \in \Delta \in S_n^\theta$ and let $\mathcal{M}, s \Vdash \Delta$ for some CGM $\mathcal{M}$ and state $s \in \mathcal{M}$. Let, furthermore, $\sigma_{A_p} \in D_{A_p}(s)$ be an A_p-move witnessing the truth of $\langle\langle A_p\rangle\rangle\bigcirc\varphi_p$ at s and let $\chi \in \mathsf{ecl}(\theta)$ be a β-formula, whose β_i-associate ($i \in \{1, 2\}$) is χ_i. Then, there exists in $\mathcal{T}_n^\theta$ an outcome set X_{χ_i} of $\sigma_{A_p}[\langle\langle A_p\rangle\rangle\bigcirc\varphi_p]$ such that for every $\Delta' \in X_{\chi_i}$ there exists $s' \in out(s, \sigma_{A_p})$ such that $\mathcal{M}, s' \Vdash \Delta'$, and moreover, if $\mathcal{M}, s' \Vdash \chi_i$, then $\chi_i \in \Delta'$.*

PROOF. Construct X_{χ_i} in a way X was constructed in the proof of the preceding lemma, with a single modification: when dealing with the formula χ, instead of choosing arbitrarily between χ_1 and χ_2, choose χ_i whenever it is true at s'. □

LEMMA 5.8. *Let $\neg\langle\langle A'_q\rangle\rangle\bigcirc\psi_q \in \Delta \in S_n^\theta$ and let $\mathcal{M}, s \Vdash \Delta$ for some CGM $\mathcal{M}$ and state $s \in \mathcal{M}$. Let, furthermore, $\sigma^c_{A'_q} \in D^c_{A'_q}(s)$ be a co-A'_q-move witnessing the truth of $\neg\langle\langle A'_q\rangle\rangle\bigcirc\psi_q$ at s. Then, there exists in $\mathcal{T}_n^\theta$ an outcome set X of $\sigma^c_{A'_q}[\neg\langle\langle A'_q\rangle\rangle\bigcirc\psi_q]$ such that for each $\Delta' \in X$ there exists $s' \in out(s, \sigma^c_{A'_q})$ such that $\mathcal{M}, s' \Vdash \Delta'$.*

PROOF. Consider the set of prestates $Y = \{\,\Gamma \in \textbf{prestates}(\Delta) \mid \Delta \xrightarrow{\sigma} \Gamma, \sigma = \sigma^c_{A'_q}[\neg\langle\langle A'_q\rangle\rangle\bigcirc\psi_q](\sigma_{A'_q})$ for some $\sigma_{A'_q} \in D_{A'_q}(\Delta)\,\}$. Take an arbitrary $\Gamma \in Y$. It follows immediately from the **(Next)** rule (see Remark 4.3) that Γ (which must contain $\neg\psi_q$) is either of the form $\{\varphi_1, \ldots, \varphi_m, \neg\psi_q\}$, where

$$\{\langle\langle A_1\rangle\rangle \bigcirc \varphi_1, \ldots, \langle\langle A_m\rangle\rangle \bigcirc \varphi_m, \neg\langle\langle A'_q\rangle\rangle \bigcirc \psi_q\} \subseteq \Delta$$

satisfies the condition of Lemma 5.2, or of the form $\{\neg\psi_q\}$.

As $\mathcal{M}, s \Vdash \Delta$, in the former case, by Lemma 5.2, there exists $s' \in out(s, \sigma^c_{A'_q})$ with $\mathcal{M}, s' \Vdash \Gamma$. Then Γ can be extended to a downward saturated set Δ' containing at least one next-time formula ($\langle\langle\Sigma_\theta\rangle\rangle \bigcirc \top$ if nothing else) such that $\mathcal{M}, s' \Vdash \Delta'$. This is done by choosing, for every β-formula to be dealt with, the "disjunct" that is actually true in $\mathcal{M}$ at s' (if both "disjuncts" are true, choose arbitrarily).

In the latter case, the same conclusion follows from Lemma 5.2 again, by adding to Δ the valid formula $\langle\langle\emptyset\rangle\rangle \bigcirc \top$.

To complete the proof of the lemma, take X to be the set of all tableau states Δ' obtainable from the prestates in Y in the way described above. □

Corollary 5.9. *Let $\neg\langle\langle A'_q\rangle\rangle \bigcirc \psi_q \in \Delta \in S^\theta_n$ and let $\mathcal{M}, s \Vdash \Delta$ for some CGM $\mathcal{M}$ and state $s \in \mathcal{M}$. Let, furthermore, $\sigma^c_{A'_q} \in D^c_{A'_q}(s)$ be a co-A'_q-move witnessing the truth of $\neg\langle\langle A'_q\rangle\rangle \bigcirc \psi_q$ at s and let $\chi \in \mathsf{ecl}(\theta)$ be a β-formula, whose β_i-associate ($i \in \{1, 2\}$) is χ_i. Then, there exists in $\mathcal{T}^\theta_n$ an outcome set X_{χ_i} of $\sigma^c_{A'_q}[\neg\langle\langle A'_q\rangle\rangle \bigcirc \psi_q]$ such that for every $\Delta' \in X_{\chi_i}$ there exists $s' \in out(s, \sigma^c_{A'_q})$ such that $\mathcal{M}, s' \Vdash \Delta'$, and moreover, if $\mathcal{M}, s' \Vdash \chi_i$, then $\chi_i \in \Delta'$.*

Proof. Analogous to the proof of Corollary 5.7. □

We now show that the tableau moves (for eventualities of the form $\langle\langle A\rangle\rangle \varphi \mathcal{U} \psi$) and co-moves (for eventualities of the form $\neg\langle\langle A\rangle\rangle \Box \varphi$) whose existence was established in the preceding two lemmas can be stitched together into what we call eventuality realization witness trees[10]. Theses trees, as already mentioned, simulate trees of runs effected in models by (co-)strategies. It will then follow that the existence of such a tree for a state Δ means that it cannot be removed from a tableau due to **(E3)**.

Definition 5.10. Let $\mathcal{R} = (R, \rightarrow)$ be a tree and X be a non-empty set. An X-*coloring* of $\mathcal{R}$ is a mapping $c : R \mapsto X$. When such mapping is fixed, we say that $\mathcal{R}$ is X-*colored.*

Definition 5.11. A *realization witness tree for the eventuality $\langle\langle A\rangle\rangle \varphi \mathcal{U} \psi$ at state* $\Delta \in S^\theta_n$ is a finite S^θ_n-colored tree $\mathcal{R} = (R, \rightarrow)$ such that

(1) the root of $\mathcal{R}$ is colored with Δ;
(2) if an interior node of $\mathcal{R}$ is colored with Δ', then $\{\varphi, \langle\langle A\rangle\rangle \bigcirc \langle\langle A\rangle\rangle \varphi \mathcal{U} \psi, \langle\langle A\rangle\rangle \varphi \mathcal{U} \psi\} \subseteq \Delta'$;
(3) for every interior node w of $\mathcal{R}$ colored with Δ', the children of w are colored bijectively with the states from an outcome set of $\sigma_A[\langle\langle A\rangle\rangle \bigcirc \langle\langle A\rangle\rangle \varphi \mathcal{U} \psi] \in D_A(\Delta')$;
(4) if a leaf of $\mathcal{R}$ is colored with Δ', then $\{\psi, \langle\langle A\rangle\rangle \varphi \mathcal{U} \psi\} \subseteq \Delta'$.

[10]In the context of this paper, by a tree we mean any directed, connected, and acyclic graph, each node of which, except one, the root, has exactly one incoming edge.

Definition 5.12. A *realization witness tree for the eventuality* $\neg\langle\langle A\rangle\rangle\Box\varphi$ *at state* $\Delta \in S_n^\theta$ is a finite S_n^θ-colored tree $\mathcal{R} = (R, \rightarrow)$ such that

(1) the root of $\mathcal{R}$ is colored with Δ;
(2) if an interior node of $\mathcal{R}$ is colored with Δ', then $\{\neg\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\Box\varphi, \neg\langle\langle A\rangle\rangle\Box\varphi\} \subseteq \Delta'$;
(3) for every interior node w of $\mathcal{R}$ colored with Δ', the children of w are colored bijectively with the states from an outcome set of $\sigma_A^c[\neg\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\Box\varphi]$;
(4) if a leaf of $\mathcal{R}$ is colored with Δ', then $\{\neg\varphi, \neg\langle\langle A\rangle\rangle\Box\varphi\} \subseteq \Delta'$.

Lemma 5.13. *Let* $\mathcal{R} = (R, \rightarrow)$ *be a realization witness tree for an eventuality* ξ *at* $\Delta \in S_n^\theta$. *Then,* ξ *is realized in* $\mathcal{T}_n^\theta$ *at every* Δ' *coloring a node of* R*—in particular, at* Δ *in* $\mathcal{T}_n^\theta$.

Proof. Straightforward induction on the length of the longest path from a node colored by Δ' to a leaf of $\mathcal{R}$ (recall that realization of eventualities was defined in Definitions 4.7 and 4.8). $\Box$

We now prove the existance of realization witness trees for satisfiable states of tableaux containing eventualities.

Lemma 5.14. *Let* $\xi \in \Delta$ *be an eventuality formula and* $\Delta \in S_0^\theta$ *be satisfiable. Then there exists a realization witness tree* $\mathcal{R} = (R, \rightarrow)$ *for* ξ *at* $\Delta \in S_0^\theta$. *Moreover, every* Δ' *coloring a node of* R *is satisfiable.*

Proof. We only supply the full proof for eventualities of the form $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$; we then indicate how to obtain the proof for eventualities of the form $\neg\langle\langle A\rangle\rangle\Box\varphi$.

If $\psi \in \Delta$, then we are done straight off—the realization witness tree is made up of a single node, the root, colored with Δ. Hence, we only need to consider the case when $\psi \notin \Delta$. As Δ is downward saturated, then $\{\varphi, \langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi\} \subseteq \Delta$.

So, suppose that $\mathcal{M}, s \Vdash \Delta$; in particular, $\mathcal{M}, s \Vdash \varphi$ and $\mathcal{M}, s \Vdash \langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$. The latter means that there exists $\sigma_A \in D_A(s)$ such that $s' \in out(s, \sigma_A)$ implies $\mathcal{M}, s' \Vdash \langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$. Now, $\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$ is a positive next-time formula. Since Δ is satisfiable, it does not contain a patent inconsistency; hence, the **(Next)** rule has been applied to it. As part of that application, $\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$ has been assigned a place, say p, in the linear ordering of the next-time formulae of Δ. Furthermore, $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$ is a β-formula whose β_2 is ψ. Therefore, Corollary 5.7 is applicable to Δ, $\chi = \langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$, $\chi_1 = \langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$, and $\chi_2 = \psi$. According to that corollary, there exists an outcome set X_ψ of $\sigma_A[\langle\langle A\rangle\rangle\bigcirc\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi]$ at Δ such that, for every $\Delta' \in X_\psi$, there exists $s' \in out(s, \sigma_A)$ such that $\mathcal{M}, s' \Vdash \Delta'$ and, moreover, if $\mathcal{M}, s' \Vdash \psi$, then $\psi \in \Delta'$. We start building the witness tree $\mathcal{R}$ by constructing a simple tree (i.e., one with a single interior node, the root) whose root r is colored with Δ and whose leaves are colored, in the way prescribed by Definition 5.11, by the states from X_ψ.

Next, since $\mathcal{M}, s' \Vdash \langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$ for every $s' \in out(s, \sigma_A)$, it follows that for every such s' there exists a (perfect-recall) A-strategy $F_A^{s'}$ such that for every $\lambda \in out(s', F_A^{s'})$ there exists $i \geq 0$ with $\mathcal{M}, \lambda[i] \Vdash \psi$ and $\mathcal{M}, \lambda[j] \Vdash \varphi$ holds for all $0 \leq j < i$. Then, playing σ_A followed by playing $F_A^{s'}$ for the $s' \in out(s, \sigma_A)$ "chosen" by the counter-coalition $\Sigma_\theta \setminus A$ constitutes a (perfect-recall) strategy F_A witnessing the truth of $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$ at s.

We, then, continue the construction of $\mathcal{R}$ as follows. For every $s' \in out(s, \sigma_A)$ (each such s' has been matched by a node of $\mathcal{R}$ at the initial stage of the construction of $\mathcal{R}$), we follow the (perfect-recall) strategy $F_A^{s'}$, matching every state s'' appearing as part of a run compliant with $F_A^{s'}$ and satisfying the requirement that $\mathcal{M}, s'' \not\Vdash \psi$ with a node w'' of $\mathcal{R}$ and matching every A-move of $F_A^{s'}$ at s'' with the A-move in the tableau $\sigma_A[\langle\langle A\rangle\rangle \bigcirc \langle\langle A\rangle\rangle \varphi \mathcal{U} \psi] \in D_A(\Delta'')$ for the state Δ'' coloring the node w''. In this way, we follow each $F_A^{s'}$ along each run, up to the point when we reach a state t where ψ is true; at that point we reach the leaf of the respective branch of the tree we are building, as by construction, the node associated with t will be colored with a state containing both ψ and $\langle\langle A\rangle\rangle \varphi \mathcal{U} \psi$.

In the manner outlined above, we are guaranteed to build a tree satisfying all conditions of Definition 5.11. Indeed, the very way the tree is built guarantees that conditions (1-4) of that definition hold. As for finiteness, assuming that the resultant tree is infinite implies that it contains an infinite branch colored with sets not containing ψ, which in turn implies the existence of $\lambda \in out(s, F_A)$ such that for every $i \geq 0$ we have $\mathcal{M}, \lambda[i] \Vdash \neg\psi$, which contradicts the fact that F_A is a strategy witnessing the truth of $\langle\langle A\rangle\rangle \varphi \mathcal{U} \psi$ at s.

Thus, we have obtained a realization witness tree $\mathcal{R}$ for $\langle\langle A\rangle\rangle \varphi \mathcal{U} \psi$ at Δ in $\mathcal{T}_n^\theta$. Moreover, it is clear from the way this tree has been built that every state coloring a node of $\mathcal{R}$ is satisfiable (in $\mathcal{M}$).

The proof for eventualities of the form $\neg\langle\langle A\rangle\rangle \Box \varphi$ is completely analogous, with reference to Corollary 5.9 rather than Corollary 5.7, using the fact that $\neg\langle\langle A\rangle\rangle \Box \varphi$ is a β-formula, with $\beta_1 = \neg\varphi$ and $\beta_2 = \neg\langle\langle A\rangle\rangle \bigcirc \langle\langle A\rangle\rangle \Box \varphi$. □

THEOREM 5.15 SOUNDNESS. *If θ is satisfiable, then $\mathcal{T}^\theta$ is open.*

PROOF. We will prove that no satisfiable states are eliminated in the state elimination phase of the construction of the tableau. The statement of the lemma will then follow immediately from Lemma 5.1, which implies that if the initial prestate $\{\theta\}$ is satisfiable, then at least one state of $\mathcal{T}^\theta$ containing θ is also satisfiable.

As the elimination process proceeds in stages, we will prove by induction on the number n of stages that, for every $\Delta \in S_0^\theta$, if Δ is satisfiable, then Δ will not be eliminated at stage n.

The base case is trivial: when $n = 0$, no eliminations have yet been done, hence no satisfiable Δ has been eliminated.

Now inductively assume that, if $\Delta' \in S_0^\theta$ is satisfiable, it has not been eliminated during the previous n stages of the elimination phase, and thus $\Delta' \in S_n^\theta$. Consider stage $n+1$ and a satisfiable $\Delta \in S_0^\theta$. By inductive hypothesis, $\Delta \in S_n^\theta$. We will now show that no elimination rule allows elimination of Δ from $\mathcal{T}_n^\theta$; hence, Δ will remain in $\mathcal{T}_{n+1}^\theta$.

(E1) As Δ is satisfiable, it clearly cannot contain both φ and $\neg\varphi$; therefore, it cannot be eliminated from $\mathcal{T}_n^\theta$ due to **(E1)**.

(E2) Due to the form of the **(Next)**-rule (see Remark 4.3), it immediately follows from Lemma 5.2 that if Δ is satisfiable, then all the prestates in **prestates**(Δ) are satisfiable, too. By virtue of Lemma 5.1, $\mathcal{T}_0^\theta$ contains for every $\sigma \in D(\Delta)$ at least one satisfiable Δ' with $\Delta \xrightarrow{\sigma} \Delta'$. By the inductive hypothesis, all such Δ' belong to $\mathcal{T}_n^\theta$; thus, Δ can not be eliminated from $\mathcal{T}_n^\theta$ due to **(E2)**.

(E3) We need to show that if Δ is satisfiable and contains an eventuality ξ, then ξ is realized at Δ in $\mathcal{T}^\theta_n$.

According to Lemma 5.14, there exists a realization witness tree $\mathcal{R} = (R, \rightarrow)$ for ξ at Δ in $\mathcal{T}^\theta_0$ and every Δ' coloring a node of R is satisfiable. Therefore, by inductive hypothesis, each such Δ' belongs to S^θ_n. Then, it is clear from the construction of $\mathcal{R}$ in the proof of Lemma 5.14, that $\mathcal{R}$ will still be a realization witness tree for Δ in $\mathcal{T}^\theta_n$. Then, by virtue of Lemma 5.13, ξ is realized at Δ in $\mathcal{T}^\theta_n$, hence cannot be eliminated due to **(E3)**. □

5.2 Completeness

Completeness of a tableau procedure means that if the final tableau for the input formula θ is open, than θ is satisfiable. The completeness proof presented in this section boils down to building a Hintikka structure $\mathcal{H}_\theta$ for the input formula θ out of the open tableau $\mathcal{T}^\theta$. Theorem 3.9 then guarantees the existence of a model for θ.

Our construction of a Hintikka structure $\mathcal{H}_\theta$ for θ out of $\mathcal{T}^\theta$ is going to resemble building a house, when bricks are assembled into prefab blocks that are then assembled into walls that are finally assembled into a complete structure. We will use analogues of all of those in our producing a Hintikka structure for θ. Larger and larger components of our construction will satisfy more and more conditions required of Hintikka sturctures by Definition 3.2, so that by the end, we are going to get a fully-fledged Hintikka structure.

The "bricks" of $\mathcal{H}_\theta$ are going to be states of $\mathcal{T}^\theta$. Being downward-saturated sets containing no patent inconsistencies (otherwise, they would have been eliminated due to **(E1)**), they satisfy conditions (H1)–(H3) of Definition 3.2.

The "prefab blocks" are going to be *locally consistent simple $\mathcal{T}^\theta$-trees*, which it is our next task to define. Intuitively, these trees are one-step components of the Hintikka structure we are building.

Definition 5.16. Let $\mathcal{W} = (W, \rightsquigarrow)$ be a tree and Y be a non-empty set. A *Y-labeling* of $\mathcal{W}$ is a mapping l from the set of edges of $\mathcal{W}$ to the set of non-empty subsets of Y. When such mapping is fixed, we say that $\mathcal{W}$ is *labeled by Y*.

Definition 5.17. A tree $\mathcal{W} = (W, \rightsquigarrow)$ is a *$\mathcal{T}^\theta$-tree* if the following conditions hold:

— $\mathcal{W}$ is S^θ-colored (recall Definition 5.10), by some coloring mapping c;
— $\mathcal{W}$ is labeled by $\cup_{\Delta \in S^\theta} D(\Delta)$, by some labeling mapping l;
— $l(w \rightsquigarrow w') \subseteq D(\Delta)$ for every $w \in W$ with $c(w) = \Delta$.

Definition 5.18. A $\mathcal{T}^\theta$-tree $\mathcal{W} = (W, \rightsquigarrow)$ is *locally consistent* if the following condition holds:

> For every interior node $w \in W$ with $c(w) = \Delta$ and every Δ-successor $\Delta' \in S^\theta$, there exists exactly one $w' \in W$ such that $l(w \rightsquigarrow w') = \{\sigma \mid \Delta \xrightarrow{\sigma} \Delta'\}$.

That is, a locally consistent tree can not have two distinct successors $w' = c(\Delta')$ and $w'' = c(\Delta'')$ of an interior node $w = c(\Delta)$ such that $\{\sigma \mid \Delta \xrightarrow{\sigma} \Delta'\} = \{\sigma \mid \Delta \xrightarrow{\sigma} \Delta''\}$. Note that we label edges of $\mathcal{T}^\theta$-trees with sets of move vectors as each edge in a tableau can be marked by more than one move vector.

Remark 5.19. Note that by construction of the tableau, different successor states Δ' of Δ are reachable from Δ by pairwise disjoint sets of moves, i.e. if, for some Δ', we have $Y = \{\,\sigma \mid \Delta \stackrel{\sigma}{\longrightarrow} \Delta'\,\}$, then if any $\sigma \in Y$ connects Δ to some other successor state Δ'', then $\{\,\sigma \mid \Delta \stackrel{\sigma}{\longrightarrow} \Delta'\,\} = Y$.

Definition 5.20. A tree $\mathcal{W} = (W, \rightsquigarrow)$ is *simple* if it has no interior nodes other than the root.

Locally consistent simple $\mathcal{T}^\theta$-trees will be our building blocks for the construction of a Hintikka structure from an open tableau $\mathcal{T}^\theta$. Essentially, we are extracting from tableaux one-step structures that resemble CGFs in that every interior node of these structures has exactly one outcome state associated with a given move vector. In other words, while an open tableau encodes all possible Hintikka structures for the input formula, we are extracting only one of them, by choosing the outcome states associated with move vectors at each state out of possibly several such outcomes.

We now prove the existence of locally consistent simple $\mathcal{T}^\theta$-trees associated with each state Δ.

Definition 5.21. Let $\Delta \in S^\theta$. A $\mathcal{T}^\theta$-tree $\mathcal{W}$ is *rooted at* Δ if the root of $\mathcal{W}$ is colored with Δ, i.e., $c(r) = \Delta$, where r is the root of $\mathcal{W}$.

Lemma 5.22. *Let $\Delta \in S^\theta$. Then, there exists a locally consistent simple $\mathcal{T}^\theta$-tree rooted at Δ.*

Proof. Such a tree can be built as follows: consider all successor states Δ' of Δ in $\mathcal{T}^\theta$. With each of them is associated a non-empty set of "move vectors" $\{\,\sigma \mid \Delta \stackrel{\sigma}{\longrightarrow} \Delta'\,\}$. The $\mathcal{T}^\theta$-tree will then consist of a root r colored with Δ and a leaf associated with each such set of move vectors (see Remark 5.19), colored with any of the successor states Δ' with which this particular set of moves is associated (note that, in general, a tableau can contain more than one such Δ'); the edge between the root r and a leaf t is then labeled by the set of moves $\{\,\sigma \mid \Delta \stackrel{\sigma}{\longrightarrow} c(t)\,\}$. □

The next lemma essentially asserts that, in addition to conditions (H1)–(H3), locally consistent simple $\mathcal{T}^\theta$-trees also satisfy conditions (H4)–(H5) of Definition 3.2, where outcomes of A-moves and co-A-moves are defined for such trees as for CGFs; recall that edges of these trees are labeled with sets of move vectors. Thus, locally consistent simple $\mathcal{T}^\theta$-trees are closely approximating Hintikka structures, but so far only *locally*.

Prior to proving the lemma, we extend the meaning of the notation introduced in Notational convention 5.5.

Notational convention 5.23. Let $\mathcal{W} = (W, \rightsquigarrow)$ be $\mathcal{T}^\theta$-tree and let $\Delta = c(w)$ for some $w \in W$. As before, $\langle\langle A_p \rangle\rangle \bigcirc \varphi_p \in \Delta$ means that $\langle\langle A_p \rangle\rangle \bigcirc \varphi_p$ is the pth positive next-time formula in the list produces as part of applying **(Next)** to Δ; $\neg\langle\langle A'_q \rangle\rangle \bigcirc \psi_q \in \Delta$ has similar meaning.

(1) Given $\langle\langle A_p \rangle\rangle \bigcirc \varphi_p \in \Delta$, by $\sigma_{A_p}[\langle\langle A_p \rangle\rangle \bigcirc \varphi_p]$ we denote (the unique) A_p-move $\sigma_{A_p} \in D_{A_p}(w)$ such that $\sigma_{A_p}(a) = p$ for every $a \in A_p$.

(2) Given a proper $\neg\langle\langle A'_q \rangle\rangle \bigcirc \psi_q \in \Delta \in S^\theta_n$, by $\sigma^c_{A'_q}[\neg\langle\langle A'_q \rangle\rangle \bigcirc \psi_q]$ we denote (the unique) co-A'_q-move satisfying the following condition: $\mathbf{neg}(\sigma^c_{A'_q}(\sigma_{A'_q})) = q$ and $\Sigma_\theta \setminus A'_q \subseteq N(\sigma^c_{A'_q}(\sigma_{A'_q}))$ for every $\sigma_{A'_q} \in D_{A'_q}(w)$.

LEMMA 5.24. *Let $\mathcal{S}$ be a locally consistent simple $\mathcal{T}^\theta$-tree rooted at Δ. Then, the following hold:*

(1) *If $\langle\!\langle A\rangle\!\rangle\bigcirc\varphi \in \Delta = c(w)$, then there exists an A-move $\sigma_A \in D_A(w)$ such that $\varphi \in \Delta'$ for all $\Delta' = c(w')$, where $w' \in out(w, \sigma_A)$.*

(2) *If $\neg\langle\!\langle A\rangle\!\rangle\bigcirc\varphi \in \Delta = c(w)$, then there exists a co-A-move $\sigma^c_A \in D^c_A(w)$, such that $\neg\varphi \in \Delta'$ for all $\Delta' = c(w')$, where $w' \in out(w, \sigma^c_A)$.*

PROOF. Note that every $\Delta \in S^\theta$ is not patently inconsistent. Therefore, we can assume throughout the proof that all next-time formulae of Δ have been linearly ordered as part of applying the **(Next)** rule to Δ.

(1) Suppose that $\langle\!\langle A\rangle\!\rangle\bigcirc\varphi \in \Delta$. Then the required A-move is $\sigma_A[\langle\!\langle A\rangle\!\rangle\bigcirc\varphi]$. Indeed, it immediately follows from the rule **(Next)** that for every $\sigma \sqsupseteq \sigma_A$ in the pretableau $\mathcal{P}^\theta$, if $\Delta \xrightarrow{\sigma} \Gamma$ then $\varphi \in \Gamma$. Now, in $\mathcal{T}^\theta$ we have $\Delta \xrightarrow{\sigma} \Delta'$ only if in $\mathcal{P}^\theta$ we had $\Delta \xrightarrow{\sigma} \Gamma$ for some $\Gamma \subseteq \Delta'$. Therefore, $\varphi \in \Delta'$ for every Δ' in any outcome set of σ_A at Δ, and the statement of the lemma follows.

(2) Suppose that $\neg\langle\!\langle A\rangle\!\rangle\bigcirc\varphi \in \Delta$. We have two cases to consider.

Case 1: $A \neq \Sigma_\theta$. Therefore, there exists $b \in \Sigma_\theta \setminus A$ and, furthermore, $\neg\langle\!\langle A\rangle\!\rangle\bigcirc\varphi$ occupies some place, say q, in the linear ordering of the next-time formulae of Δ. Consider an arbitrary $\sigma_A \in D_A(w)$. We claim that σ_A can be extended to $\sigma' \sqsupseteq \sigma_A$ such that in the final tableau we have $\Delta \xrightarrow{\sigma'} \Delta'$ and $\neg\varphi \in \Delta'$ for some Δ'. To show that, denote by $N(\sigma_A)$ the set $\{\, i \mid \sigma_A(i) \geq m \,\}$, where m is the number of positive next-time formulae in Δ, and by $\mathbf{neg}(\sigma_A)$ the number $\left(\sum_{i\in N(\sigma_A)}(\sigma_A(i) - m)\right)$ mod l, where l is the number of negative next-time formulae in Δ. Now, consider $\sigma' \sqsupseteq \sigma_A$ defined as follows: $\sigma'_b = ((q - \mathbf{neg}(\sigma_A)) \bmod l) + m$ and $\sigma'_{a'} = m$ for any $a' \in \Sigma_\theta \setminus (A \cup \{b\})$. It is easy to see that $\Sigma_\theta \setminus A \subseteq N(\sigma')$, and moreover, that $\mathbf{neg}(\sigma') = (\mathbf{neg}(\sigma_A) + (q - \mathbf{neg}(\sigma_A))) \bmod l = q$. We conclude that in the pretableau $\mathcal{P}^\theta$, if $\Delta \xrightarrow{\sigma'} \Gamma$, then $\neg\varphi \in \Gamma$. But $\mathcal{S}$ contains at least one leaf colored with such Δ' that $\Delta \xrightarrow{\sigma'} \Delta'$, and this Δ' was obtained by extending a Γ with $\Delta \xrightarrow{\sigma'} \Gamma$; hence, $\neg\varphi \in \Delta'$, and the statement of the lemma follows.

Case 2: $A = \Sigma_\theta$. Then, by virtue of (H2), $\langle\!\langle \emptyset\rangle\!\rangle\bigcirc\neg\varphi \in \Delta$ and thus, by the rule **(Next)**, $\neg\varphi \in \Gamma$ for every $\Gamma \in \mathbf{prestates}(\Delta)$. Then, $\neg\varphi \in \Delta'$ for every Δ' that is a successor of Δ in $\mathcal{T}^\theta$ and hence in the coloring set of every leaf of $\mathcal{S}$. Then, the (unique) co-Σ_θ-move, which is an identity function, has the required properties. □

Now, we come to the "walls" of our building—the components of the future Hintikka structure that take care of single eventualities. Following [Goranko and van Drimmelen 2006], we call them *final tree components.* Each final tree component is built around a realization witness tree for the corresponding eventuality (recall Definitions 5.11 and 5.12), the existence of which is proved in the forthcoming lemma.

LEMMA 5.25. *Let ξ be an eventuality realized at Δ in $\mathcal{T}^\theta_n$. Then, there exists a realization witness tree $\mathcal{R}$ for ξ at Δ in $\mathcal{T}^\theta_n$.*

PROOF. To build $\mathcal{R}$, we use the concept of the realization rank of Δ in $\mathcal{T}^\theta_n$ with respect to an eventuality ξ, which we define as the shortest path from Δ to a state witnessing the realization of ξ at Δ (if $\xi = \langle\!\langle A \rangle\!\rangle \varphi \mathcal{U} \psi$, such a state contains ψ; if $\xi = \neg \langle\!\langle A \rangle\!\rangle \Box \varphi$, then such a state contains $\neg\varphi$), denoted by $\mathbf{rank}(\Delta, \xi, \mathcal{T}^\theta_n)$. If such a path does not exists, then $\mathbf{rank}(\Delta, \xi, \mathcal{T}^\theta_n) = \infty$. Clearly, if ξ is realized at Δ in $\mathcal{T}^\theta_n$, then $\mathbf{rank}(\Delta, \xi, \mathcal{T}^\theta_n)$ is finite.

Suppose, first, that ξ is of the form $\langle\!\langle A \rangle\!\rangle \varphi \mathcal{U} \psi$. We start building $\mathcal{R}$ by taking a root node and coloring it with Δ. Afterwards, for every $w' \in \mathcal{R}$ colored with Δ', we do the following: for every $\sigma \sqsupseteq \sigma_A[\langle\!\langle A \rangle\!\rangle \bigcirc \langle\!\langle A \rangle\!\rangle \varphi \mathcal{U} \psi] \in D(w')$, we pick the node w'' colored with $\Delta'' \in \mathbf{succ}_\sigma(\Delta')$ with the least $\mathbf{rank}(\Delta'', \langle\!\langle A \rangle\!\rangle \varphi \mathcal{U} \psi, \mathcal{T}^\theta_n)$ and add to $\mathcal{R}$ a child w'' of w' coloree with Δ''. As $\langle\!\langle A \rangle\!\rangle \varphi \mathcal{U} \psi$ is realized at Δ, it follows that $\mathbf{rank}(\Delta, \langle\!\langle A \rangle\!\rangle \varphi \mathcal{U} \psi, \mathcal{T}^\theta_n)$ is finite. By construction of $\mathcal{R}$ and definition of the rank, each child of every node of the so constructed tree has a smaller realization rank than the parent. Therefore, along each branch of the tree we are bound to reach in a finite number of steps a node colored with a state whose realization rank with respect to $\langle\!\langle A \rangle\!\rangle \varphi \mathcal{U} \psi$ is 0; such nodes are taken to be the leaves of $\mathcal{R}$. As every node of $\mathcal{R}$ has finitely many children, due to König's lemma, $\mathcal{R}$ is finite. Therefore, so constructed $\mathcal{R}$ is indeed a realization witness tree for $\langle\!\langle A \rangle\!\rangle \varphi \mathcal{U} \psi$ at Δ in $\mathcal{T}^\theta_n$.

Suppose, next, that ξ is of the form $\neg \langle\!\langle A \rangle\!\rangle \Box \varphi$. Again, we begin by taking a root node and coloring it with Δ. Afterwards, for every $w' \in \mathcal{R}$ colored with Δ', we do the following: for every $\sigma = \sigma^c_A[\neg \langle\!\langle A \rangle\!\rangle \bigcirc \langle\!\langle A \rangle\!\rangle \Box \varphi](\sigma_A) \in D(w')$, we pick the w'' colored with $\Delta'' \in \mathbf{succ}_\sigma(\Delta')$ with the least $\mathbf{rank}(\Delta'', \neg \langle\!\langle A \rangle\!\rangle \Box \varphi, \mathcal{T}^\theta_n)$ and add to $\mathcal{R}$ a child w'' of w' colored with Δ''. The rest of the argument is analogous to the one for the other eventuality. □

Now, we are going to use realization witness trees to build $\mathcal{T}^\theta$-trees doing the same job for eventualities as realization witness trees do, i.e., "realizing" them in a certain sense. The problem with realization witness trees is that their nodes might lack successors along some "move vectors"; the next definition and lemma show that this shortcoming can be easily remedied, by giving each interior node Δ of a realization witness tree a successor associated with every move vector $\sigma \in \Delta$.

Definition 5.26. Let $\mathcal{W} = (W, \rightsquigarrow)$ be a locally consistent $\mathcal{T}^\theta$-tree rooted at Δ and $\xi \in \Delta$ be an eventuality formula. We say that $\mathcal{W}$ *realizes* ξ if there exists a subtree[11] $\mathcal{R}_\xi$ of $\mathcal{W}$ rooted at Δ such that $\mathcal{R}_\xi$ is a realization witness tree for ξ at Δ in $\mathcal{T}^\theta$.

LEMMA 5.27. *Let $\xi \in \Delta \in S^\theta$ be an eventuality formula. Then, there exist a finite locally consistent $\mathcal{T}^\theta$-tree $\mathcal{W}_\xi$ rooted at Δ realizing ξ.*

PROOF. Take the realization witness tree $\mathcal{R}_\xi$ for ξ at Δ in $\mathcal{T}^\theta$, which exists by Lemma 5.25. The only reason why $\mathcal{R}_\xi$ may turn out not to be a locally consistent $\mathcal{T}^\theta$-tree is that some of its interior nodes do not have a successor node along every move vector σ (recall that, in realization witness trees, every interior node has just enough successors to witness realization of the corresponding eventuality, and no more). Therefore, to build a locally consistent $\mathcal{T}^\theta$-tree out of $\mathcal{R}_\xi$, we simply add to its interior nodes just enough "colored" successors so that (1) for every interior node w' of $\mathcal{W}_\xi$ and every $\sigma \in D(w')$, the tree $\mathcal{W}_\xi$ contains a w'' such that $c(w'') = \Delta''$

[11]By a subtree, we mean a graph obtained from a tree by removing some of its nodes together with all the nodes reachable from them.

for some $\Delta'' \in \mathbf{succ}_{\sigma}(\Delta')$ (where $\Delta' = c(w')$) and (2) $\mathcal{W}_{\xi}$ satisfies the condition of Definition 5.18. It is then obvious that $\mathcal{W}_{\xi}$ is a locally consistent $\mathcal{T}^{\theta}$-tree, by definition realizing ξ. Moreover, as according to Lemma 5.25, $\mathcal{R}_{\xi}$ is finite, $\mathcal{W}_{\xi}$ is finite, too. □

We want to build Hintikka structures out of locally consistent $\mathcal{T}^{\theta}$-trees. Hintikka structures are based on CGFs; therefore, we need to be able to "embed" such trees into CGFs. The following definition formally defines such an embedding.

Definition 5.28. Let $\mathcal{W} = (W, \rightsquigarrow)$ be a locally consistent $\mathcal{T}^{\theta}$-tree and $\mathfrak{F} = (\Sigma_{\theta}, S, d, \delta)$ be a CGF. We say that $\mathcal{W}$ is *contained* in $\mathfrak{F}$, denoted $\mathcal{W} \ll \mathfrak{F}$, if the following conditions hold:

— $W \subseteq S$;
— if $\sigma \in l(w \rightsquigarrow w')$, then $\delta(w, \sigma) = w'$.

Locally consistent $\mathcal{T}^{\theta}$-trees realizing an eventuality ξ are meant to represent run trees in CGFs effected by (co-)strategies. We now show that if we embed the former variety of tree into a CGF then, as expected, this gives rise to a positional (co-)strategy witnessing the truth of ξ under an "appropriate valuation". (Intuitively, this (co-)strategy is extracted out of a locally consistent $\mathcal{T}^{\theta}$-tree when it is embedded into a CGF and can, thus, be viewed as a run tree). The following two lemmas prove this for the two types of eventualities we have in the language.

LEMMA 5.29. *Let,* $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi \in \Delta \in S^{\theta}$ *and let* $\mathcal{W} = (W, \rightsquigarrow)$ *be a locally consistent* $\mathcal{T}^{\theta}$*-tree rooted at* Δ *and realizing* $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$*. Let, furthermore,* $\mathfrak{F} = (\Sigma_{\theta}, S, d, \delta)$ *be a CGF such that* $\mathcal{W} \ll \mathfrak{F}$*. Then, there exists a positional A-strategy* F_A *in* $\mathfrak{F}$ *such that, if* $\lambda \in out(w, F_A)$*, where* $c(w) = \Delta$*, then there exists* $i \geq 0$ *such that* $\psi \in \lambda[i] \in W$ *and* $\varphi \in \lambda[j] \in W$ *holds for all* $0 \leq j < i$*.*

PROOF. At every interior node w' of the realization witness tree for $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$, which is contained in $\mathcal{W}$, take the A-move $\sigma_A[\langle\langle A\rangle\rangle\bigcirc\varphi\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi]$. At any other node, for definiteness' sake, take the lexicographically first A-move. This strategy is clearly positional and has the required property. □

LEMMA 5.30. *Let,* $\neg\langle\langle A\rangle\rangle\Box\varphi \in \Delta \in S^{\theta}$ *and let* $\mathcal{W} = (W, \rightsquigarrow)$ *be a locally consistent* $\mathcal{T}^{\theta}$*-tree rooted at* Δ *and realizing* $\neg\langle\langle A\rangle\rangle\Box\varphi$*. Let, furthermore,* $\mathfrak{F} = (\Sigma_{\theta}, S, d, \delta)$ *be a CGF such that* $\mathcal{W} \ll \mathfrak{F}$*. Then, there exists a positional co-A-strategy* F^c_A *in* $\mathfrak{F}$ *such that, if* $\lambda \in out(w, F^c_A)$*, where* $c(w) = \Delta$*, then* $\neg\varphi \in \lambda[i] \in W$ *for every* $i \geq 0$*.*

PROOF. At every interior node w' of the realization witness tree for $\neg\langle\langle A\rangle\rangle\Box\varphi$, which is contained in $\mathcal{W}$, take the co-A-move $\sigma^c_A[\neg\langle\langle A\rangle\rangle\bigcirc\varphi]$. At any other node, for definiteness' sake, take the lexicographically first co-A-move. This co-A-strategy is clearly positional and has the required property. □

Our next big step in the completeness proof is to assemble locally consistent $\mathcal{T}^{\theta}$-trees realizing eventualities, as well as locally consistent simple $\mathcal{T}^{\theta}$-trees, into a Hintikka structure for θ. To do that, we need the concept of partial concurrent game frame that generalizes that of CGF. Partial CGFs are different from CGFs in that they have "deadlocked" states, i.e., states for which the transition function δ is not defined (the analog in Kripke frames would be "dead ends"—the states that cannot "see" any other states); however, each deadlocked state of a partial CGF is required to be an image of a transition function δ for some (ordinary) state.

We need partial CGFs as we will be building a Hintikka structure for θ step-by-step, all but the final step producing partial CGFs having deadlocked states that will be given successors at the next stage of the construction. Put another way, the motivation for introducing partial CGFs is that locally consistent $\mathcal{T}^\theta$-trees are partial CGFs, and we build a Hintikka structure for θ out of such trees.

Definition 5.31. A *partial concurrent game frame* (partial CGF, for short) is a tuple $\mathfrak{S} = (\Sigma, S, Q, d, \delta)$, where

— Σ is a finite, non-empty set of *agents*;

— $S \neq \emptyset$ is a set of *states*;

— $Q \subseteq S$ is a set of *deadlock states*;

— d is a function assigning to every $a \in \Sigma$ and every $s \in S \setminus Q$ a natural number $d_a(s) \geq 1$ of *moves* available to agent a at state s; notation $D_a(s)$ and $D(s)$ has the same meaning as in the case of CGFs (see Definition 2.2);

— δ is a *transition function* satisfying the following requirements:
— $\delta(s, \sigma) \in S$ for every $s \in S \setminus Q$ and every $\sigma \in D(s)$;
— for every $q \in Q$, there exist $s \in S \setminus Q$ and $\sigma \in D(s)$ such that $q = \delta(s, \sigma)$.

The concept of A-move is defined for partial CGFs in a way analogous to the way it is defined for CGFs; the only difference is that, in the former case, A-moves are only defined for states in $S \setminus Q$. The set of all A-moves at state $s \in S \setminus Q$ is denoted by $D_A(s)$. Outcomes of A-moves are defined exactly as for CGFs. Analogously for co-A-moves.

Definition 5.32. Let $\mathfrak{S} = (\Sigma, S, Q, d, \delta)$ be a partial CGF and $A \subseteq \Sigma$. A *positional A-strategy in* $\mathfrak{S}$ is a mapping $F_A : S \mapsto \bigcup \{ D_A(s) \mid s \in S \setminus Q \}$ such that $F_A(s) \in D_A(s)$ for all $s \in S \setminus Q$.

Definition 5.33. Let $\mathfrak{S} = (\Sigma, S, Q, d, \delta)$ be a partial CGF and $A \subseteq \Sigma$. A *positional co-A-strategy in* $\mathfrak{S}$ is a mapping $F^c_A : S \mapsto \bigcup \{ D^c_A(s) \mid s \in S \setminus Q \}$ such that $F^c_A(s) \in D^c_A(s)$ for all $s \in S \setminus Q$.

We now establish a fact that will be crucial to our ability to stitch partial CGFs that are locally consistent $\mathcal{T}^\theta$-trees together. Intuitively, given such a partial CGF $\mathfrak{S}$ and a state w of $\mathfrak{S}$ colored with a set Δ'' containing an eventuality $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$, coalition A has a strategy such that every (finite) run compliant with that strategy either realizes $\langle\langle A\rangle\rangle\varphi\,\mathcal{U}\psi$ or postpones its realization until a deadlocked state (Lemma 5.35). Analogously for eventualities of the form $\neg\langle\langle A\rangle\rangle\Box\varphi$ and co-A-strategies (Lemma 5.36). First, a technical definition.

Definition 5.34. Let $\mathfrak{S} = (\Sigma, S, Q, d, \delta)$ be a partial CGF and let $s \in S$. An *s-fullpath* in $\mathfrak{S}$ is a finite sequence $\rho = s_0, \ldots, s_n$ of elements of S such that

— $s_0 = s$;

— for every $0 \leq i < n$, there exists $\sigma \in D(s_i)$ such that $s_{i+1} = \delta(s_i, \sigma)$;

— $s_n \in Q$.

The fullpath $\rho = s_0, \dots, s_n$ is compliant with the strategy F_A, denoted $\rho \in out(F_A)$, if $s_{i+1} \in out(F_A(s_i))$ for all $0 \leq i < n$. Analogously for co-strategies. The length of ρ (defined as the number of positions in ρ) is denoted by $|\rho|$.

LEMMA 5.35. *Let $\mathfrak{S} = (\Sigma_\theta, S, Q, d, \delta)$ be a partial CGF such that*

(1) *S is colored with elements of S^θ;*
(2) *for every $w \in S$, the set $\{w\} \cup \{ w' \mid w' = \delta(w, \sigma),$ for some $\sigma \in D(w) \}$ is a set of nodes of a locally consistent simple $\mathcal{T}^\theta$-tree;*
(3) *$\langle\langle A \rangle\rangle \varphi \mathcal{U} \psi \in \Delta''$, where $\Delta'' = c(w'')$ for some $w'' \in S$;*

Then, there exists a positional A-strategy F_A in $\mathfrak{S}$ such that, for every w''-fullpath $\rho \in out(F_A)$, either of the following holds:

— *there exists $0 \leq i < |\rho|$ such that $\psi \in c(\rho[i])$ and $\varphi \in c(\rho[j])$ for every $0 \leq j < i$;*

— *$\varphi \in c(\rho[i])$ for every $0 \leq i < |\rho|$.*

PROOF. Straightforward. □

LEMMA 5.36. *Let $\mathfrak{S} = (\Sigma_\theta, S, Q, d, \delta)$ be a partial CGF such that*

(1) *S is colored with elements of S^θ;*
(2) *for every $w \in S$, the set $\{w\} \cup \{ w' \mid w' = \delta(w, \sigma),$ for some $\sigma \in D(w) \}$ is a set of nodes of a locally consistent simple $\mathcal{T}^\theta$-tree;*
(3) *$\neg\langle\langle A \rangle\rangle \Box \varphi \in \Delta''$, where $\Delta'' = c(w'')$ for some $w'' \in S$;*

Then, there exists a positional co-A-strategy F^c_A in $\mathfrak{S}$ such that $\neg\varphi \in c(\rho[i])$ for every Δ''-fullpath $\rho \in out(F^c_A)$ and every $i \geq 0$.

PROOF. Straightforward. □

Now, we define the building blocks, referred to as *final tree components*, from which a Hintikka structure for θ will be built; the construction is essentially taken from [Goranko and van Drimmelen 2006].

Definition 5.37. Let $\Delta \in S^\theta$ and $\xi \in \mathcal{T}^\theta$ be an eventuality formula. Then, the *final tree component for ξ and Δ*, denoted $\mathcal{F}_{(\xi,\Delta)}$, is defined as follows:

— if $\xi \in \Delta$, then $\mathcal{F}_{(\xi,\Delta)}$ is a finite locally consistent $\mathcal{T}^\theta$-tree $\mathcal{W}_\xi$ rooted at Δ realizing ξ; the existence of such a tree being guaranteed by Lemma 5.27;

— if $\xi \notin \Delta$, then $\mathcal{F}_{(\xi,\Delta)}$ is a locally consistent simple $\mathcal{T}^\theta$-tree rooted at Δ; the existence of such a tree being guaranteed by Lemma 5.22.

We are now ready to define what we will prove to be a (positional) Hintikka structure for the input formula θ, which we denote by $\mathcal{H}_\theta$. We start by defining the CGF $\mathfrak{F}$ underlying $\mathcal{H}_\theta$.

To that end, we first arrange all states of $\mathcal{T}^\theta$ in a list $\Delta_0, \dots, \Delta_{n-1}$ and all eventualities occurring in the states of $\mathcal{T}^\theta$ in a list $\xi_0, \dots, \xi_{m-1}$. We then think of all the final tree components (see Definition 5.37) as arranged in an m-by-n grid whose rows are marked with the correspondingly numbered eventualities of $\mathcal{T}^\theta$ and whose columns are marked with the correspondingly numbered states of $\mathcal{T}^\theta$. The final tree component found at the intersection of the ith row and the jth column will be denoted by $\mathcal{F}_{(i,j)}$. The building blocks for $\mathfrak{F}$ will all come from the grid, and we build $\mathfrak{F}$ incrementally, at each state of the construction producing a partial CGF realizing more and more eventualities. The crucial fact here is that if an eventuality ξ is not realized within a partial CGF used in the construction, then ξ is "passed down" to be realized later, in accordance with Lemmas 5.35 and 5.36.

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

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

After that, the cycle is repeated, but with a crucial modification that will guarantee that the CGHS we are building is going to be finite: whenever the component we are about to attach, say $\mathcal{F}_{(i,j)}$, is already contained in the partial CGF we have constructed thus far, instead of replacing the deadlocked state w (such that $c(w) = \Delta_j$) with that component, we connect every "predecessor" v of w to the root of $\mathcal{F}_{(i,j)}$ by an arrow $\rightsquigarrow$ marked with the set $l(v \rightsquigarrow w)$. This modified version of the cycle is repeated until we come to a point when no more components get added. This result in a finite CGF $\mathfrak{F}$. Now, to define $\mathcal{H}_\theta$, we simply put $H(w) = c(w)$, for every $w \in \mathfrak{F}$.

Theorem 5.38. *The above defined $\mathcal{H}_\theta$ is a (positional) Hintikka structure for θ.*

Proof. The "for θ" part immediately follows the construction of $\mathcal{H}_\theta$ (recall the very first step of the construction). It, thus, remains to argue that $\mathcal{H}_\theta$ is indeed a Hintikka structure.

Conditions (H1)–(H3) of Definition 3.2 hold since states of $\mathcal{H}_\theta$ are consistent downward saturated sets.

Conditions (H4) and (H5) essentially follow from Lemma 5.24.

Condition (H6) follows from the way $\mathcal{H}_\theta$ is constructed together with lemmas 5.29 and 5.35. Lastly, condition (H7) follows from the way $\mathcal{H}_\theta$ is constructed together with lemmas 5.30 and 5.36.

Lastly, $\mathcal{H}_\theta$ is positional by construction. Indeed, it is built from final tree components, which are locally consistent simple $\mathcal{T}^\theta$-trees; as we have seen in Lemmas 5.29 and 5.30, when embedded into CGFs, these trees give rise to positional strategies. □

The positionality of $\mathcal{H}_\theta$ gives us the following, stronger, version of the completeness theorem for our tableau procedure:

THEOREM 5.39 POSITIONAL COMPLETENESS. *Let θ be an **ATL** formula and let $\mathcal{T}^\theta$ be open. Then, θ is satisfiable in a CGM based on a frame with positional strategies.*

COROLLARY 5.40. *If an **ATL**-formula θ is tightly satisfiable, then it is tightly satisfiable in a positional CGM.*

PROOF. Suppose that θ is tightly satisfiable in a CGM based on a CGF with perfect recall strategies. Then, by Theorem 5.15, the tableau $\mathcal{T}^\theta$ for θ is open. It then follows form Theorem 5.39 that θ is satisfiable in a positional CGM. □

6. SOME VARIATIONS OF THE METHOD

In the present section, we sketch some immediate adaptations of the tableau method described above for testing other strains of satisfiability, such as loose **ATL**-satisfiability and **ATL**-satisfiability over some special classes of frames. Other, less straightforward, adaptations will be developed in follow-up work.

6.1 Loose satisfiability for **ATL**

The procedure described above is easily adaptable to testing **ATL**-formulae for loose satisfiability, which the reader will recall, is satisfiability over frames with exactly one agent not featuring in the formula. All that is necessary to adapt the above-described procedure to testing for this strain of satisfiability is the modification of the **(Next)** rule in such a way that it accommodates $|\Sigma_\theta| + 1$ agent rather than $|\Sigma_\theta|$. As such a modification is entirely straightforward, we omit the details. The complexity of the procedure is not affected.

6.2 **ATL** over special classes of frames

Some classes of concurrent game frames are of particular interest (for motivation and examples, see [Alur et al. 2002]).

6.2.1 *Turn-based synchronous frames.* In turn-based synchronous frames, at every state, exactly one agent has "real choices". Thus, agents take it in turns to act.

Definition 6.1. A concurrent game frame $\mathfrak{F} = (\Sigma, S, d, \delta)$ is *turn-based synchronous* if, for every $s \in S$, there exists agent $a_s \in \Sigma$, referred to as the *owner of* s, such that $d_a(s) = 1$ for all $a \in \Sigma \setminus \{a_s\}$.

To tests formulae for satisfiability over turn-based synchronous frames, we need to make the following adjustments to the above tableau procedure (we are assuming that we are testing for tight satisfiability; loose satisfiability is then straightforward). All the states of the tableau are now going to be "owned" by individual agents. Intuitively, if Δ is "owned" by $a \in \Sigma_\theta$, it is agent's a turn to act at Δ; we indicate ownership by affixing the name of the owner as a subscript of the state. The rule **(SR)** now looks as follows:

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

(1) for every $a \in \Sigma_\theta$, add to the pretableau all the minimal downward saturated extensions of Γ, marked with a (all thus created sets Δ_a are *a-states*);
(2) for each of the so obtained states Δ_a, if Δ_a does not contain any formulae of the form $\langle\langle A\rangle\rangle\bigcirc\varphi$ or $\neg\langle\langle A\rangle\rangle\bigcirc\varphi$, add the formula $\langle\langle\Sigma_\theta\rangle\rangle\bigcirc\top$ to Δ_a;
(3) for each state Δ_a obtained at steps 1 and 2, put $\Gamma \Longrightarrow \Delta_a$;
(4) if, however, the pretableau already contains a state Δ'_a that coincides with Δ_a, do not create another copy of Δ'_a, but only put $\Gamma \Longrightarrow \Delta'_a$.

Moreover, when creating prestates from a-states, all agents except a get exactly one vote, while a can still vote for any next-time formula in the current state. The rule **(Next)**, therefore, now looks as follows:

(Next) Given a state Δ_a such that for no χ we have $\chi, \neg\chi \in \Delta_a$, do the following:

(1) order linearly all positive and proper negative next-time formulae of Δ_a in such a way that all the positive next-time formulae precede all the negative ones; suppose the result is the list

$$\mathbb{L} = \langle\langle A_0\rangle\rangle\bigcirc\varphi_0, \ldots, \langle\langle A_{m-1}\rangle\rangle\bigcirc\varphi_{m-1}, \neg\langle\langle A'_0\rangle\rangle\bigcirc\psi'_0, \ldots, \neg\langle\langle A'_{l-1}\rangle\rangle\bigcirc\psi_{l-1}.$$

(Due to step 2 of **(SR)**, $\mathbb{L}$ is non-empty.) Let $r_\Delta = m + l$; denote by $D(\Delta_a)$ the set $\{\sigma \in \mathbb{N}^{|\Sigma_\theta|} \mid 0 \leq \sigma_a < r_\Delta \text{ and } \sigma_b = 0, \text{ for all } b \neq a\}$;

(2) consider the elements of $D(\Delta_a)$ in the lexicographic order and for each $\sigma \in D(\Delta_a)$ do the following:
   (a) create a prestate

$$\begin{aligned}\Gamma_\sigma &= \{\varphi_p \mid \langle\langle A_p\rangle\rangle\bigcirc\varphi_p \in \Delta_a \text{ and } a \in A_p \text{ and } \sigma_a = p\}\\ &\cup \{\varphi_p \mid \langle\langle A_p\rangle\rangle\bigcirc\varphi_p \in \Delta_a \text{ and } a \notin A_p\}\\ &\cup \{\neg\psi_q \mid \neg\langle\langle A'_q\rangle\rangle\bigcirc\psi_q \in \Delta_a \text{ and } a \in A'_q\}\\ &\cup \{\neg\psi_q \mid \neg\langle\langle A'_q\rangle\rangle\bigcirc\psi_q \in \Delta_a \text{ and } a \notin A'_q \text{ and } \sigma_a = q\}\end{aligned}$$

   put $\Gamma_\sigma = \{\top\}$ if all four sets above above are empty.
   (b) connect Δ_a to Γ_σ with $\stackrel{\sigma}{\longrightarrow}$;
   If, however, $\Gamma_\sigma = \Gamma$ for some prestate Γ that has already been added to the pretableau, only connect Δ to Γ with $\stackrel{\sigma}{\longrightarrow}$.

Otherwise, tableaux testing for satisfiability over turn-based synchronous frames are no different from those for satisfiability over all frames.

6.2.2 *Moore synchronous frames.* In Moore synchronous frames over the set of agents Σ, the set of states S can be represented as a Cartesian product of sets of local states $S_{a\in\Sigma}$, one for each agent. The actions of agents are determined by the current "global" state $s \in S$; each action σ_a of agent a at state $s \in S$, however, results in a local state determined by a function δ_a mapping pairs $\langle$ global state, a-move$\rangle$ into S_a. Then, given a move vector $\sigma \in D(s)$, representing simultaneous actions of all agents at s, the σ-successor of s is determined by the local states of agents produced by their actions—namely, it is a k-tuple (where $k = |\Sigma|$) of respective local states $(\delta_1(s, \sigma_1), \ldots, \delta_k(s, \sigma_k))$, one for each agent. This intuition can be formalized as follows (see [Alur et al. 2002]):

Definition 6.2. A CGF $\mathfrak{F} = (\Sigma, S, d, \delta)$ is *Moore synchronous* if the following two conditions are satisfied, where $k = |\Sigma|$:

— $S = S_1 \times \cdots \times S_k$;

— for each state $s \in S$, move vector σ, and agent $a \in \Sigma$, there exists a local state $\delta_a(s, \sigma_a)$ such that $\delta(s, \sigma) = (\delta_1(s, \sigma_1), \ldots, \delta_k(s, \sigma_k))$.

Definition 6.3. A CGF $\mathfrak{F} = (\Sigma, S, d, \delta)$ is *bijective*, if $\delta(s, \sigma) \neq \delta(s, \sigma')$ for every $s \in S$ and every σ and σ' such that $\sigma \neq \sigma'$.

It is easy to see that every bijective frame is isomorphic to a Moore synchronous one. Therefore, if—for whatever reason—using our tableau procedure, we want to produce a Moore synchronous model for the input formula, we simply never identify the states created in the course of applying the **(Next)** rule. This clearly produces a bijective, and hence Moore synchronous, model. By inspecting the tableau procedure, it can be noted that identification or otherwise of the states never affects the output of the procedure. Therefore, an analysis of our tableau procedure leads to the following claim:

THEOREM 6.4 [GORANKO 2001]. *Let θ be an **ATL**-formula. Then, θ is satisfiable in the class of all CGFs iff it is satisfiable in the class of Moore synchronous CGFs.*

7. CONCLUDING REMARKS

We have developed a complexity-efficient terminating incremental-tableau-based decision procedure for **ATL** and some of its variations. This style of tableaux for **ATL**, while having the same worst-case upper bound as the other known decision procedures, including the top-down tableaux-like procedure presented in [Walther et al. 2006], is expected to perform better in practice because, as it has been shown in the examples, it creates much fewer tableau states.

We believe that the tableau method developed herein is not only of more immediate practical use, but also is more flexible and adaptable than any of the decision procedures developed earlier in [van Drimmelen 2003], [Goranko and van Drimmelen 2006], and [Walther et al. 2006]. In particular, this method can be suitably adapted to variations of **ATL** with committed strategies [Ågotnes et al. 2007] and **ATL** with incomplete information, which is the subject of follow-up work.

ACKNOWLEDGMENTS

This research was supported by a research grant of the National Research Foundation of South Africa and was done during the second author's post-doctoral fellowship in School of Mathematics at the University of the Witwatersrand, funded by the Claude Harris Leon Foundation. We gratefully acknowledge the financial support from these institutions. We also gratefully acknowledge the detailed and useful comments of the referees, which have helped us significantly improve the presentation of the paper.

Received March 2008; accepted August 2008

REFERENCES

ABATE, P., GORÈ, R., AND WIDMANN, F. 2007. One-pass tableaux for Computation Tree Logic. In *Lecture Notes in Computer Science*. Springer-Verlag, 32–46. Proc. LPAR 2007.

ÅGOTNES, T., GORANKO, V., AND JAMROGA, W. 2007. Alternating-time temporal logics with irrevocable strategies. In *Proceedings of the 11th International Conference on Theoretical Aspects of Rationality and Knowledge (TARK XI)*, D. Samet, Ed. Presses Universitaires de Louvain, Univ. Saint-Louis, Brussels, 15–24.

ALUR, R., HENZINGER, T. A., AND KUPERMAN, O. 1997. Alternating-time temporal logic. In *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science*. 100–109.

ALUR, R., HENZINGER, T. A., AND KUPERMAN, O. 1998. Alternating-time temporal logic. In *Lecture Notes in Computer Science*. Vol. 1536. Springer-Verlag, 23–60.

ALUR, R., HENZINGER, T. A., AND KUPERMAN, O. 2002. Alternating-time temporal logic. *Journal of the ACM 49,* 5, 672–713.

ALUR, R., HENZINGER, T. A., MANG, F. Y. C., QADEER, S., RAJAMANI, S. K., AND TASIRAN, S. 1998. Mocha: Modularity in model-checking. In *Lecture Notes in Computer Science*. Vol. 1427. Springer-Verlag, 521–525.

BRADFIELD, J. AND STIRLING, C. 2007. Modal μ-calculi. In *Handbook of Modal Logic*, P. Blackburn, Ed. Elsevier, 721–756.

EMERSON, E. A. 1990. Temporal and modal logics. In *Handbook of Theoretical Computer Science*, J. van Leeuwen, Ed. Vol. B. MIT Press, 995–1072.

EMERSON, E. A. AND HALPERN, J. 1985. Decision procedures and expressiveness in the temporal logic of branching time. *Journal of Computation and System Sciences 30,* 1, 1–24.

FAGIN, R., HALPERN, J. Y., MOSES, Y., AND VARDI, M. Y. 1995. *Reasoning about Knowledge*. MIT Press: Cambridge, MA.

FITTING, M. C. 1983. *Proof Methods for Modal and Intuitionistic Logic*. D. Reidel Publishing Co.

FITTING, M. C. 2007. Modal proof theory. In *Handbook of Modal Logic*, P. Blackburn, J. van Bentem, and F. Wolter, Eds. Elsevier, 85–138.

GORANKO, V. 2001. Coalition games and alternating temporal logics. In *Proceedings of the 8th conference on Theoretical Aspects of Rationality and Knowledge (TARK VIII)*, J. van Benthem, Ed. Morgan Kaufmann, 259–272.

GORANKO, V. AND JAMROGA, W. 2004. Comparing semantics of logics for multi-agent systems. *Synthese 139,* 2, 241–280.

GORANKO, V. AND SHKATOV, D. Tableau-based decision procedures for logics of strategic ability in multi-agent systems. To appear in *ACM Transactions on Computational Logic*. Available at `http://arxfiv.org/abs/0803.2306`.

GORANKO, V. AND SHKATOV, D. 2008. Tableau-based decision procedure for the multi-agent epistemic logic with operators of common and distributed knowledge. In *Proc. of the 6th IEEE Iternational Conference on Software Engineering and Formal Methods*. IEEE Press. Available at `http://arxiv.org/abs/0808.4133`.

GORANKO, V. AND VAN DRIMMELEN, G. 2006. Complete axiomatization and decidablity of Alternating-time temporal logic. *Theoretical Computer Science 353*, 93–117.

GORÉ, R. 1988. Tableau methods for modal and temporal logics. In *Handbook of Tableau Methods*, M. D'Agostino, D. Gabbay, and R. Hänhle, Eds. Kluwer.

HANSEN, H. H. 2004. Tableau games for Coalition Logic and Alternating-time Temporal Logic. M.S. thesis, University of Amsterdam.

HEWITT, C. 1990. The challenge of open systems. In *The Foundations of Artificial Intelligence – a Sourcebook*, D. Partridge and Y. Wilks, Eds. Cambridge University Press, 383–395.

Marx, M., Mikulás, S., and Reynolds, M. 2000. The mosaic method for temporal logics. In *Lecture Notes in Computer Science*. Vol. 1847. Springer-Verlag, 324–340.

Pauly, M. 2001a. Logic for Social Software. Ph.D. thesis, University of Amsterdam. ILLC Dissertation Series 2001-10.

Pauly, M. 2001b. A logical framework for coalitional effectivity in dynamic procedures. *Bulletin of Economic Research 53,* 4 (October), 305–324.

Pauly, M. 2002. A modal logic for coalitional power in games. *Journal of Logic and Computation 12,* 1 (February), 149–166.

Pauly, M. and Parikh, R. 2003. Game logic—an overview. *Studia Logica 75,* 2, 165–182.

Shoham, Y. and Leyton-Brown, K. 2008. *Multiagent Systems: Algorithmic, Game-Theoretic and Logical Foundations*. CUP.

Smullyan, R. M. 1968. *First-order Logic*. Springer-Verlag.

Thomas, W. 1995. On the synthesis of strategies in infinite games. In *Proceedings of the 12th Annual Symposium on Theoretical Aspects of Computer Science, STACS '95*, E. Mayr and C. Puech, Eds. Vol. LNCS 900. Springer, 1–13.

van Drimmelen, G. 2003. Satisfiability in alternating-time temporal logic. In *Proceedings of 18th IEEE Symposium on Logic in Computer Science (LICS)*. 208–217.

Walther, D., Lutz, C., Wolter, F., and Wooldridge, M. 2006. ATL satisfiability is indeed ExpTime-complete. *Journal of Logic and Computation 16,* 6, 765–787.

Weiss, G., Ed. 1999. *Multiagent Systems*. MIT Press.

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

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