# Model Checking Temporal-Epistemic Logic Using Tree Automata

Francesco Belardinelli, Andrew V. Jones, and Alessio Lomuscio

Department of Computing
Imperial College London
London, UK

{f.belardinelli,andrew.jones,a.lomuscio}@imperial.ac.uk

**Abstract.** We introduce an automata-theoretic approach for the verification of multi-agent systems. We present a translation between branching time temporal-epistemic logic and alternating tree automata. Model checking an interpreted system against a temporal-epistemic formula is reduced to checking the non-emptiness of the composition of two tree automata. We exemplify this technique using a simple multi-agent scenario.

## 1 Introduction

Multi-agent systems (MAS) have arisen as a useful paradigm for reasoning about distributed systems. In such systems, multiple autonomous agents engage in social activities, such as learning or co-operation, with the aim of achieving their desired goal. Therefore, when reasoning about multi-agent systems, we are not only interested in how the system evolves over time, but also about what the agents in these systems can know. This desire has lead to the application of modal logics for time and knowledge (temporal-epistemic logic) to MAS [2].

The formal reasoning of agents has lead to the development of automatic techniques for the verification of multi-agent systems against temporal-epistemic specifications. Unfortunately, the inherently loose-coupled nature of MAS means that the state-space explosion problem for these systems is even more acute. Symbolic model checking approaches have therefore been seen to be a preferential approach to ameliorating the state-space explosion problem [5,9].

In contrast to symbolic approaches, orthogonal techniques for the temporal-only reasoning of systems focuses on the use of language acceptance of automata. If the model is accepted by the corresponding automaton for a formula, then the model satisfies the formula. When dealing with branching time logics, alternating tree automata are used that allow for the reasoning about multiple successors of a state [3]. The application of branching time automata-theoretic approaches to temporal-epistemic logic has yet to be investigated.

In this paper we propose an automata-theoretic approach to verifying the branching time temporal-epistemic logic CTLK (Section 2). Part of the approach involves unravelling both the temporal and the epistemic successors of a model into distinct successors of its corresponding tree (Section 3). Translating temporal-epistemic properties into tree automata requires the matching of successor states

based upon the current sub-formulae being translated, e.g., when translating an epistemic sub-formula, it is only necessary to consider epistemic successors (Section 4). We prove that the product automata of a formula and a model is empty if and only if the model satisfies the formula (Section 5). We also mention further applications of the technique developed (Section 6).

### 1.1 Related Work

Our work directly extends the branching time automata-theoretic approach of Kupferman *et al.* [3] to support epistemic transitions. The application of tree automata to branching time model checking has seen considerable previous research, focusing on different approaches including those such as "amorphous" tree automata [1].

Existing approaches to agent verification either verify a restricted class of agent systems, or are based on symbolic approaches. Automata-theoretic verification of MAS has been applied to the class of interpreted systems supporting "perfect recall" [11]. In this class of systems indistinguishability is based on traces and verification can take place over word automata. Conversely, our technique supports indistinguishability between global states and knowledge is interpreted as an multi-modal S5 operator.

There have been two main avenues of research into the symbolic verification of agent systems. The approach of Lomuscio *et al.* [5] involves the use of BDDs for efficient manipulation of Boolean formulae representing sets of states in the model. Penczek *et al.* [9] apply bounded model checking (BMC) to epistemic logic; their work presents a breadth-first translation of interpreted systems models and CTLK formulae to the Boolean satisfiability problem (SAT). The approach we present is more tailored towards explicit-state model checking where the states of the system are explicitly represented. This is a common division between symbolic and automata-theoretic approaches.

## 2 Prerequisites

In this section we introduce the temporal-epistemic logic CTLK, which combines the computation tree logic CTL with the multi-modal epistemic logic $\mathrm{S5}_m$ for a set $A = \{1, \ldots, m\}$ of agents. Then we provide this language with a formal semantics by means of interpreted systems, which is the typical semantic framework for temporal-epistemic logics in multi-agent systems [6,2].

### 2.1 The Temporal Epistemic Logic CTLK

Given a set $P = \{p_1, p_2, \ldots\}$ of propositional variables, the temporal-epistemic language $\mathcal{L}_m$ contains the propositional variables in $P$, the connectives $\neg$ and $\rightarrow$, the linear time operators $X$ and $U$, the branching time operators $A$ and $E$, and the epistemic operator $K_i$ for each agent $i \in A$.

**Definition 1.** *The formulae in $\mathcal{L}_m$ are defined in BNF as follows:*

$$\phi ::= p \mid \neg\phi \mid \phi \rightarrow \phi \mid AX\phi \mid A\phi U\phi \mid E\phi U\phi \mid K_i\phi$$

The formulae $AX\phi$ and $A\phi U\phi'$ (resp. $E\phi U\phi'$) are read as "*in all paths, at the next step* $\phi$" and "*in all paths (resp. for some path), eventually* $\phi'$ *and until then* $\phi$". The formula $K_i\phi$ represents "*agent* $i$ *knows* $\phi$". We define the connectives $\wedge$, $\vee$, $\leftrightarrow$ as standard. The operator $X$ is self-dual, that is, $EX\phi$ is defined as $\neg AX\neg\phi$. The linear time operator $\overline{U}$ is dual to $U$, that is, $A\phi\overline{U}\phi'$ is defined as $\neg E\neg\phi U\neg\phi'$, and $E\phi\overline{U}\phi'$ as $\neg A\neg\phi U\neg\phi'$. The operator $\overline{U}$ is also referred to as the *release* operator $R$. The epistemic possibility $\overline{K}_i$ is dual to $K_i$ for each $i \in A$, that is, $\overline{K}_i\phi$ is defined as $\neg K_i\neg\phi$. Also, the operators $AG$, $AF$, $EG$ and $EF$ are defined as standard.

The $U$-formulae in $\mathcal{L}_m$ are the formulae of the form $A\phi U\phi'$ or $E\phi U\phi'$ for some $\phi$, $\phi' \in \mathcal{L}_m$; the $\overline{U}$- and $K_i$-formulae are defined similarly.

### 2.2 Interpreted Systems

We introduce interpreted systems by assuming a set $L_i$ of local states $l_i, l_i', \ldots$ for each agent $i \in A$ in a multi-agent system, as well as the environment $e$. The set $\mathcal{S} \subseteq L_e \times L_1 \times \ldots \times L_m$ contains the global states in the multi-agent system. To represent the temporal evolution of the MAS we consider the flow of time $\mathbb{N}$ of the natural numbers. A *run* in an interpreted system is a function $\rho : \mathbb{N} \rightarrow \mathcal{S}$. Intuitively, a run represents a possible evolution of the MAS assuming $\mathbb{N}$ as the flow of time. Finally, we define the interpreted systems for the language $\mathcal{L}_m$ as follows.

**Definition 2 (IS).** *An* interpreted system *is a tuple* $\mathcal{P} = \langle \mathcal{R}, s^0, I \rangle$ *where*

*(i)* $\mathcal{R}$ *is a non-empty set of runs;*
*(ii)* $s^0 \in \mathcal{S}$ *is the initial state;*
*(iii)* $I : \mathcal{S} \rightarrow 2^P$ *is an assignment for the propositional variables in* $P$*.*

*We denote by* $\mathcal{IS}_m$ *the class of interpreted systems with* $m$ *agents.*

Note that in general, in any interpreted system $\mathcal{P}$, we have $\{s \in \mathcal{S} \mid s = \rho(n)$ for some $\rho \in \mathcal{R}$, $n \in \mathbb{N}\} \subseteq \mathcal{S}$, but the converse is not always true. In what follows we assume without loss of generality that for every $s \in \mathcal{S}$, there exist $\rho \in \mathcal{R}$ and $n \in \mathbb{N}$ such that $s = \rho(n)$.

Following standard notation [2] a pair $(\rho, n)$ is a *point* in $\mathcal{P}$. If $\rho(n) = \langle l_e, l_1, \ldots, l_m \rangle$ is the global state at point $(\rho, n)$ then $\rho_e(n) = l_e$ and $\rho_i(n) = l_i$ are the environment's and agent $i$'s local state at $(\rho, n)$ respectively. Further, for $i \in A$ the equivalence relation $\sim_i$ is defined such that $(\rho, n) \sim_i (\rho', n')$ only if $\rho_i(n) = \rho_i'(n')$. We denote by $[(\rho, n)]_{\sim_i}$ the equivalence class $\{(\rho', n') \mid (\rho, n) \sim_i (\rho', n')\}$.

Now we assign a meaning to the formulae in $\mathcal{L}_m$ by means of interpreted systems.

**Definition 3.** *The satisfaction relation* $\models$ *for* $\phi \in \mathcal{L}_m$ *and* $(\rho, n) \in \mathcal{P}$ *is defined as follows:*

$(\mathcal{P}, \rho, n) \models p$ *iff* $p \in I(\rho(n))$
$(\mathcal{P}, \rho, n) \models \neg\psi$ *iff* $(\mathcal{P}, \rho, n) \not\models \psi$
$(\mathcal{P}, \rho, n) \models \psi \rightarrow \psi'$ *iff* $(\mathcal{P}, \rho, n) \not\models \psi$ *or* $(\mathcal{P}, \rho, n) \models \psi'$
$(\mathcal{P}, \rho, n) \models AX\psi$ *iff for all runs* $\rho'$, $\rho'(n') = \rho(n)$ *implies* $(\mathcal{P}, \rho', n'+1) \models \psi$
$(\mathcal{P}, \rho, n) \models A\psi U\psi'$ *iff for all runs* $\rho'$, $\rho'(n') = \rho(n)$ *implies that there is* $n'' \geq n'$, $(\mathcal{P}, \rho', n'') \models \psi'$ *and* $n' \leq n''' < n''$ *implies* $(\mathcal{P}, \rho', n''') \models \psi$
$(\mathcal{P}, \rho, n) \models E\psi U\psi'$ *iff for some run* $\rho'$, $\rho'(n') = \rho(n)$ *and there is* $n'' \geq n'$, $(\mathcal{P}, \rho', n'') \models \psi'$ *and* $n' \leq n''' < n''$ *implies* $(\mathcal{P}, \rho', n''') \models \psi$
$(\mathcal{P}, \rho, n) \models K_i\psi$ *iff* $(\rho, n) \sim_i (\rho', n')$ *implies* $(\mathcal{P}, \rho', n') \models \psi$

The truth conditions for $\wedge$, $\vee$, $\leftrightarrow$, $EX$, $A\overline{U}$, $E\overline{U}$ and $\overline{K}_i$ are defined from those above. A formula $\phi$ is *true on a IS* $\mathcal{P}$ iff it is satisfied at $s^0$; $\phi$ is *valid on a class* $\mathcal{C}$ *of IS* iff it is true on every IS in $\mathcal{C}$.

## 3 Alternating Tree Automata

In this section we present alternating tree automata, which generalise nondeterministic automata. These were first introduced in [8] and have been used in [3] to define an automata-theoretic technique to model check branching time logics.

Let $A_t$ be the set $A \cup \{t\}$ containing all the agents in $A$ plus the temporal index $t$. Hence, $|A_t| = m + 1$.

**Definition 4 ($A_t$-tree).** *An $A_t$-tree is a set $T \subseteq (\mathbb{N} \times A_t)^*$ such that if $x \cdot (c, j) \in T$ for $x \in (\mathbb{N} \times A_t)^*$ and $(c, j) \in \mathbb{N} \times A_t$, then*

- $x \in T$;
- *for all* $0 \leq c' < c$, *also* $x \cdot (c', j) \in T$.

The elements of $T$ are called *nodes*; the empty word $\epsilon$ is the root of $T$. For every $x \in T$, the nodes $x \cdot (c, j)$ are the *$j$-successors* of $x$. The number of $j$-successors of $x$ is called the *$j$-degree* of $x$ and is denoted by $d_j(x)$; the vector of all successor degrees of $x$ is denoted by $\boldsymbol{d}(x)$. A *leaf* is a node with no successors.

**Definition 5 (path).** *A* path *in a tree $T$ is a set $\pi \subseteq T$ such that*

- $\epsilon \in \pi$,
- *for every $x \in \pi$, either $x$ is a leaf or there exists a unique $(c, j) \in \mathbb{N} \times A_t$ such that $x \cdot (c, j) \in \pi$.*

*A* temporal *path $\pi$ is a path where $j = t$.*

For any path $\pi$, $n \in \mathbb{N}$, $\pi^n$ represents the $n$-th element in the path.

Given an alphabet $\Sigma$, a $\Sigma$-labelled tree is a pair $\langle T, V \rangle$ where $T$ is a tree and $V : T \rightarrow \Sigma$ maps each node of $T$ to a letter in $\Sigma$. Note that an infinite word in $\Sigma^\omega$ can be viewed as a $\Sigma$-labelled tree in which $|A_t| = 1$ and the degree of all nodes is 1.

We focus on $\Sigma$-labelled trees in which $\Sigma = \mathcal{S}$ for some set $\mathcal{S}$ of global states or $\Sigma = 2^P$; so $V$ can intuitively be seen as an assignment of propositional variables to nodes. Given an interpreted system $\mathcal{P} = \langle \mathcal{R}, s^0, I \rangle$ we can define a tree $\langle T_\mathcal{P}, V_\mathcal{P} \rangle$ with $\Sigma = \mathcal{S}$ such that $V_\mathcal{P}(\epsilon) = s^0$ and

1. if $V_{\mathcal{P}}(x) = s$ and $s_0, \ldots, s_k$ is an enumeration of all $s'$ such that $s' \sim_i s$ for $i \in A$, then $V_{\mathcal{P}}(x \cdot (c, i)) = s_c$ for $0 \leq c \leq k$;
2. if $V_{\mathcal{P}}(x) = s$ and $s_0, \ldots, s_k$ is an enumeration of all $s'$ such that there exists $\rho \in \mathcal{R}$, $n \in \mathbb{N}$ such that $\rho(n) = s$ and $\rho(n+1) = s'$, then $V_{\mathcal{P}}(x \cdot (c, t)) = s_c$ for $0 \leq c \leq k$.

Furthermore, given a tree $\langle T, V \rangle$ with $\Sigma = 2^P$ we can define a satisfaction relation $\models$ for $\phi \in \mathcal{L}_m$, $x \in T$ and a temporal path $\pi \subseteq T$ as follows:

$(T, x) \models p$ iff $p \in V(x)$
$(T, x) \models \neg\psi$ iff $(T, x) \not\models \psi$
$(T, x) \models \psi \rightarrow \psi'$ iff $(T, x) \not\models \psi$ or $(T, x) \models \psi'$
$(T, x) \models AX\psi$ iff for all temporal paths $\pi$, if $\pi^0 = x$ then $(T, \pi^1) \models \psi$
$(T, x) \models A\psi U\psi'$ iff for all temporal paths $\pi$, if $\pi^0 = x$ then there is $n \geq 0$ such that $(T, \pi^n) \models \psi'$ and $0 \leq n' < n$ implies $(T, \pi^{n'}) \models \psi$
$(T, x) \models E\psi U\psi'$ iff for some temporal path $\pi$, $\pi^0 = x$ and there is $n \geq 0$ such that $(T, \pi^n) \models \psi'$ and $0 \leq n' < n$ implies $(T, \pi^{n'}) \models \psi$
$(T, x) \models K_i\psi$ iff for all $0 \leq c < d_i(x)$, $(T, x \cdot (c, i)) \models \psi$

Given an IS $\mathcal{P}$, there is a tree $\langle T_{\mathcal{P}}, V_{\mathcal{P}} \rangle$ with $\Sigma = \mathcal{S}$. If we identify each $s \in \mathcal{S}$ with $\{p \in P \mid p \in I(s)\} \in 2^P$, then $\langle T_{\mathcal{P}}, V_{\mathcal{P}} \rangle$ is also a tree with $\Sigma = 2^P$, and we can prove the following result:

**Lemma 1.** *For every $\phi \in \mathcal{L}_m$, for $V_{\mathcal{P}}(x) = \rho(n)$, $(T_{\mathcal{P}}, x) \models \phi$ iff $(\mathcal{P}, \rho, n) \models \phi$*

In particular we have that $(T_{\mathcal{P}}, \epsilon) \models \phi$ iff $\phi$ is true in the IS $\mathcal{P}$. In what follows we will focus on the class $\mathcal{T}$ of trees $\langle T_{\mathcal{P}}, V_{\mathcal{P}} \rangle$ for some interpreted system $\mathcal{P}$.

### 3.1 Example 1 – Unwinding an Interpreted System

A simple interpreted system with a single agent $i$ is shown in Figure 1. We use solid lines to represent temporal transitions and dashed lines to represent epistemically indistinguishable states for agent $i$. In Section 4.1 we assume that each state is labelled with the atomic proposition $p$.

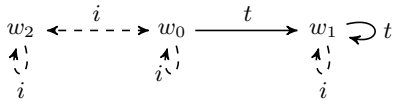


**Fig. 1.** An example interpreted system.

Figure 2 shows the $\mathcal{S}$-labelled tree $\langle T, V \rangle$ unwinding of Figure 1. Each node in the tree of Figure 2, of the form $x, V(x)$, represents the node of the tree $T$ (i.e., $x$) along with the mapping in $V$ of that node to a state in $\mathcal{S}$ (i.e., $V(x) \in \mathcal{S}$). The interpreted system of Figure 1 is cyclic (i.e., it contains reflexive loops), therefore its corresponding unwinding is an infinite tree. We only show a truncated part of the tree, the infinite part of the tree is represented with dotted lines.

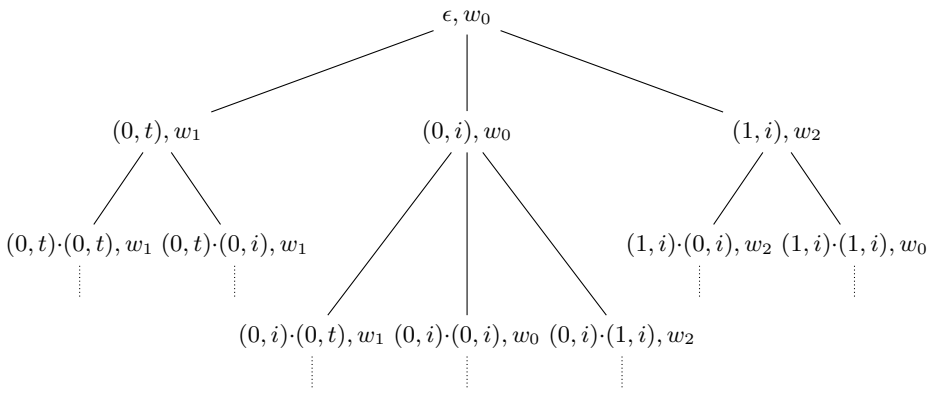


**Fig. 2.** The tree automata "unwinding" of the interpreted system in Figure 1.

We now introduce alternating tree automata. Given a set $\mathcal{D} \subset \mathbb{N}$, a $\mathcal{D}$-tree is an $A_t$-tree in which all the nodes have degrees in $\mathcal{D}$. Further, $\mathcal{B}^+(P)$ is the set of positive Boolean formulae over the set $P$ of propositional variables. For instance, $\mathcal{B}^+(\{p, q\})$ includes $p \wedge q$, $p \vee q$, $p \wedge p$.

**Definition 6 (Alternating tree automaton).** *An alternating tree automaton is a tuple $\mathcal{A} = \langle \Sigma, \mathcal{D}, Q, \delta, q_0, A_t, F \rangle$ such that*

- *$\Sigma$, $\mathcal{D}$ and $A_t$ are defined as above;*
- *$Q$ is a set of states;*
- *$q_0 \in Q$ is the initial state;*
- *$F \subseteq Q$ is the set of final states;*
- *$\delta : Q \times \Sigma \times \mathcal{D}^{|A_t|} \to \mathcal{B}^+(\mathbb{N} \times A_t \times Q)$ is the transition function.*

*Also, we require that if the atom $(c, j, q')$ appears in $\delta(q, \sigma, k_t, k_1, \ldots, k_m)$ then $0 \leq c < k_j$.*

When the automaton is in state $q$ as it reads a node that is labelled by a letter $\sigma$ and has $k_j$ $j$-successors, it applies the transition $\delta(q, \sigma, k_t, k_1, \ldots, k_m)$. In what follows we denote the tuple $\langle k_t, k_1, \ldots, k_m \rangle$ as $\boldsymbol{k}$.

A run of an alternating tree automaton $\mathcal{A}$ over a tree $\langle T, V \rangle$ is a tree $\langle T_r, r \rangle$ in which the root is labelled by $(\epsilon, q_0)$ and every other node is labelled by an element of $(\mathbb{N} \times A_t)^* \times Q$. Each node of $T_r$ corresponds to a node of $T$. On the other hand, many nodes of $T_r$ can correspond to the same node of $T$. Formally, a run $\langle T_r, r \rangle$ is a $\Sigma_r$-labelled tree where $\Sigma_r = (\mathbb{N} \times A_t)^* \times Q$ and $\langle T_r, r \rangle$ satisfies the following:

1. $r(\epsilon) = (\epsilon, q_0)$
2. Let $y \in T_r$ with $r(y) = (x, q)$. If $\delta(q, V(x), \boldsymbol{d}(x)) = \theta$ then there are (possibly empty) sets $S_j = \{(c_0, j, q_0), \ldots, (c_n, j, q_n)\} \subseteq \{0, \ldots, d_j(x) - 1\} \times \{j\} \times Q$ such that the following hold:
   - the assignment which assigns $\top$ to all the atoms in $\bigcup_{j \in A_t} S_j$ satisfies $\theta$,
   - for all $0 \leq i < n$, we have $y \cdot (i, j) \in T_r$ and $r(y \cdot (i, j)) = (x \cdot (c_i, j), q_i)$.

Note that if, for some $y$, the transition function $\delta$ has value $\top$, then $y$ need not have successors. Also, $\delta$ can never have the value $\bot$ in a run.

A run $\langle T_r, r\rangle$ is *accepting* if all its infinite paths satisfy the acceptance condition. Here we consider a Büchi acceptance condition. Given a run $\langle T_r, r\rangle$ and an infinite path $\pi \subseteq T_r$, let $inf(\pi) \subseteq Q$ be the set of $q \in Q$ such that there are infinitely many $y \in \pi$ for which $r(y) \in (\mathbb{N} \times A_t)^* \times \{q\}$. That is, $inf(\pi)$ contains exactly all the states that appear infinitely often in $\pi$. The acceptance condition is defined as follows:

- A path $\pi$ satisfies a Büchi acceptance condition $F \subseteq Q$ if and only if $inf(\pi) \cap F \neq \emptyset$.

An automaton accepts a tree if and only if there exists a run that accepts it. We denote by $\mathcal{L}(\mathcal{A})$ the set of all $\Sigma$-labelled trees that $\mathcal{A}$ accepts. Note that an alternating automaton over infinite words is simply an alternating automaton over infinite trees with $\mathcal{D} = \{1\}$ and $|A_t| = 1$. Formally, we define an alternating automaton over infinite words as $\mathcal{A} = \langle \Sigma, Q, \delta, q_0, F\rangle$ where $\delta : Q \times \Sigma \to \mathcal{B}^+(Q)$.

The model checking procedure for CTLK considers weak alternating automata (WAAs), a particular class of automata first introduced in [7].

**Definition 7.** *A WAA is an alternating tree automaton with a Büchi acceptance condition $F \subseteq Q$. Further, there is a partition of $Q$ into disjoint sets $Q_1, \ldots, Q_n$ such that for each $Q_i$, either $Q_i \subseteq F$, in which case $Q_i$ is an accepting set, or $Q_i \cap F = \emptyset$, in which case $Q_i$ is a rejecting set. In addition, there is a partial order $\leq$ on the collection of the $Q_i$s such that for every $q \in Q_i$ and $q' \in Q_j$ for which $q'$ occurs in $\delta(q, \sigma, \boldsymbol{k})$ for some $\sigma \in \Sigma$, $\boldsymbol{k} \in \mathcal{D}^{|A_t|}$, we have $Q_j \leq Q_i$.*

Thus, transitions from a state in $Q_i$ lead to states in either the same $Q_i$ or a lower one. It follows that every infinite path of a run of a WAA ultimately gets trapped within some $Q_i$. The path then satisfies the acceptance condition if and only if $Q_i$ is an accepting set. We call the partition of $Q$ into sets the *weakness partition* and we call the partial order over the sets of the weakness partition the *weakness order*.

## 4 Model Checking for CTLK

In this section we provide the construction of a weak alternating automaton $\mathcal{A}_{\mathcal{D},\psi}$ that accepts all and only the $\mathcal{D}$-trees in $\mathcal{T}$ satisfying a given CTLK formula $\psi \in \mathcal{L}_m$. In the next section the automaton $\mathcal{A}_{\mathcal{D},\psi}$ will be used to construct the product word automaton $\mathcal{A}_{\mathcal{P},\psi}$ for a given IS $\mathcal{P}$. Then we will prove that the language $\mathcal{L}(\mathcal{A}_{\mathcal{P},\psi})$ is non-empty iff the tree $\langle T_{\mathcal{P}}, V_{\mathcal{P}}\rangle$ is accepted by $\mathcal{A}_{\mathcal{D},\psi}$, i.e., iff $\psi$ is true in $\mathcal{P}$. This is the theoretical background for model checking a CTLK formula $\psi$ in a IS $\mathcal{P}$. By Theorems 3.1 and 4.7 in [3] we know that all these steps can be performed in linear time.

First, we remark that by using de Morgan's laws and the definitions of operators $\overline{U}$ and $\overline{K}_i$, we can draw the negation inwards, such that it applies only to propositional variables.

Further, we define the closure $cl(\psi)$ of a formula $\psi \in \mathcal{L}_m$ as follows:

- $\psi \in cl(\psi)$;
- if $\neg\phi \in cl(\psi)$ then $\phi \in cl(\psi)$;
- if $\phi \to \phi' \in cl(\psi)$ then $\phi, \phi' \in cl(\psi)$;
- if $AX\phi \in cl(\psi)$ or $EX\phi \in cl(\psi)$ then $\phi \in cl(\psi)$;
- if $A\phi U\phi' \in cl(\psi)$ or $E\phi U\phi' \in cl(\psi)$ then $\phi, \phi' \in cl(\psi)$;
- if $A\phi\overline{U}\phi' \in cl(\psi)$ or $E\phi\overline{U}\phi' \in cl(\psi)$ then $\phi, \phi' \in cl(\psi)$;
- if $K_i\phi \in cl(\psi)$ or $\overline{K}_i\phi \in cl(\psi)$ then $\phi \in cl(\psi)$;

**Theorem 1.** *Given a CTLK formula $\psi \in \mathcal{L}_m$ and a set $\mathcal{D} \subset \mathbb{N}$, we can construct in linear time a WAA $\mathcal{A}_{\mathcal{D},\psi} = \langle 2^P, \mathcal{D}, cl(\psi), \delta, \psi, A_t, F\rangle$ such that the $\mathcal{D}$-tree $\langle T_\mathcal{P}, V_\mathcal{P}\rangle$ is in $\mathcal{L}(\mathcal{A}_{\mathcal{D},\psi})$ iff $\psi$ is true in $\mathcal{P}$.*

**Proof.** The set $F$ of accepting states consists of all the $\overline{U}$-formulae and $K_i$-formulae in $cl(\psi)$. It remains to define the transition function $\delta$. For all $\sigma \in 2^P$ and $\boldsymbol{k} \in \mathcal{D}^{|A_t|}$ we define:

$$
\begin{aligned}
\delta\left(p, \sigma, \boldsymbol{k}\right) &= \top \text{ if } p \in \sigma \\
\delta\left(p, \sigma, \boldsymbol{k}\right) &= \bot \text{ if } p \notin \sigma \\
\delta\left(\neg p, \sigma, \boldsymbol{k}\right) &= \top \text{ if } p \notin \sigma \\
\delta\left(\neg p, \sigma, \boldsymbol{k}\right) &= \bot \text{ if } p \in \sigma \\
\delta\left(\phi_1 \star \phi_2, \sigma, \boldsymbol{k}\right) &= \delta\left(\phi_1, \sigma, \boldsymbol{k}\right) \star \delta\left(\phi_2, \sigma, \boldsymbol{k}\right), \text{ for } \star \in \{\wedge, \vee\} \\
\delta\left(AX\phi, \sigma, \boldsymbol{k}\right) &= \bigwedge_{c=0}^{k_t-1} (c, t, \phi) \\
\delta\left(EX\phi, \sigma, \boldsymbol{k}\right) &= \bigvee_{c=0}^{k_t-1} (c, t, \phi) \\
\delta\left(A\phi_1 U\phi_2, \sigma, \boldsymbol{k}\right) &= \delta\left(\phi_2, \sigma, \boldsymbol{t}\right) \vee \left(\delta\left(\phi_1, \sigma, \boldsymbol{k}\right) \wedge \bigwedge_{c=0}^{k_t-1} (c, t, A\phi_1 U\phi_2)\right) \\
\delta\left(E\phi_1 U\phi_2, \sigma, \boldsymbol{k}\right) &= \delta\left(\phi_2, \sigma, \boldsymbol{k}\right) \vee \left(\delta\left(\phi_1, \sigma, \boldsymbol{k}\right) \wedge \bigvee_{c=0}^{k_t-1} (c, t, E\phi_1 U\phi_2)\right) \\
\delta\left(A\phi_1\overline{U}\phi_2, \sigma, \boldsymbol{k}\right) &= \delta\left(\phi_2, \sigma, \boldsymbol{k}\right) \wedge \left(\delta\left(\phi_1, \sigma, \boldsymbol{k}\right) \vee \bigwedge_{c=0}^{k_t-1} \left(c, t, A\phi_1\overline{U}\phi_2\right)\right) \\
\delta\left(E\phi_1\overline{U}\phi_2, \sigma, \boldsymbol{k}\right) &= \delta\left(\phi_2, \sigma, \boldsymbol{k}\right) \wedge \left(\delta\left(\phi_1, \sigma, \boldsymbol{k}\right) \vee \bigvee_{c=0}^{k_t-1} \left(c, t, E\phi_1\overline{U}\phi_2\right)\right) \\
\delta\left(K_i\phi, \sigma, \boldsymbol{k}\right) &= \bigwedge_{c=0}^{k_i-1} (c, i, \phi) \wedge \bigwedge_{c=0}^{k_i-1} (c, i, K_i\phi) \\
\delta\left(\overline{K}_i\phi, \sigma, \boldsymbol{k}\right) &= \bigvee_{c=0}^{k_i-1} (c, i, \phi)
\end{aligned}
$$

The weakness partition and order of $\mathcal{A}_{\mathcal{D},\psi}$ are defined as follows. Each formula $\phi \in cl(\psi)$ constitutes a (singleton) set $\{\phi\}$ in the partition. The partial order is defined by $\{\phi_1\} \leq \{\phi_2\}$ iff $\phi_1 \in cl(\phi_2)$. Since each transition of the automaton from a state $\phi$ leads to states associated with formulae in $cl(\phi)$, the weakness conditions hold. In particular, each set is either contained in $F$ or disjoint from $F$.

We now prove the correctness of our construction. By Lemma 1 the formula $\psi$ is true in $\mathcal{P}$ iff $\langle T_\mathcal{P}, \epsilon\rangle \models \psi$. So it is left to prove that the $\mathcal{D}$-tree $\langle T_\mathcal{P}, V_\mathcal{P}\rangle$ is

in $\mathcal{L}(\mathcal{A}_{\mathcal{D},\psi})$ iff $\langle T_{\mathcal{P}}, \epsilon\rangle \models \psi$. We first prove that $\mathcal{A}_{\mathcal{D},\psi}$ is sound. That is, given an accepting run $\langle T_r, r\rangle$ of $\mathcal{A}_{\mathcal{D},\psi}$ over the tree $\langle T_{\mathcal{P}}, V_{\mathcal{P}}\rangle$, we prove that for every $y \in T_r$ such that $r(y) = (x, \phi)$ we have that $\langle T_{\mathcal{P}}, x\rangle \models \phi$. Thus, in particular, $\langle T_{\mathcal{P}}, \epsilon\rangle \models \psi$. The proof proceeds by induction on the structure of $\phi$. If $\phi$ is an atomic proposition and $r(y) = (x, p)$ then $\delta(p, V_{\mathcal{P}}(x), \boldsymbol{d}(x)) = \top$ iff $p \in V_{\mathcal{P}}(x)$, i.e., iff $\langle T_{\mathcal{P}}, x\rangle \models \phi$. The cases where $\phi$ is $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $AX\phi_1$, or $EX\phi_1$ follow easily, by the induction hypothesis, from the definition of $\delta$. If $\phi$ is $K_i\phi_1$ and $r(y) = (x, K_i\phi_1)$ then $\delta(K_i\phi_1, V_{\mathcal{P}}(x), \boldsymbol{d}(x)) = \bigwedge_{c=0}^{d_i(x)-1}(c, i, \phi_1) \wedge \bigwedge_{c=0}^{d_i(x)-1}(c, i, K_i\phi_1)$. Thus, for all $0 \leq k < 2d_i(x) - 1$, we have $y \cdot (k, i) \in T_r$. Also, for $0 \leq k < d_i(x)$ and $c = k$, $r(y \cdot (k, i)) = (x \cdot (c, i), \phi_1)$, and for $d_i(x) \leq k < 2d_i(x) - 1$, $c = k - d_i(x)$, $r(y \cdot (k, i)) = (x \cdot (c, i), K_i\phi_1)$. By induction hypothesis, $\langle T_{\mathcal{P}}, x \cdot (c, i)\rangle \models \phi_1$ for $0 \leq c < d_i(x)$, and therefore $\langle T_{\mathcal{P}}, x\rangle \models K_i\phi_1$. Consider now the case where $\phi$ is of the form $A\phi_1 U\phi_2$ or $E\phi_1 U\phi_2$. As $\langle T_r, r\rangle$ is an accepting run, it visits the state $\phi$ only finitely often. Since $\mathcal{A}_{\mathcal{D},\psi}$ keeps inheriting $\phi$ as long as $\phi_2$ is not satisfied, then it is guaranteed, by the definition of $\delta$ and the induction hypothesis, that along all paths or some path, as required in $\phi$, $\phi_2$ does eventually holds and $\phi_1$ holds until then. Finally, consider the case where $\phi$ is of the form $A\phi_1\overline{U}\phi_2$ or $E\phi_1\overline{U}\phi_2$. Here, it is guaranteed, by the definition of $\delta$ and the induction hypothesis, that $\phi_2$ holds either always or until both $\phi_2$ and $\phi_1$ hold.

We now prove that $\mathcal{A}_{\mathcal{D},\psi}$ is complete, that is, if $\langle T_{\mathcal{P}}, V_{\mathcal{P}}\rangle$ is a $\mathcal{D}$-tree such that $\langle T_{\mathcal{P}}, \epsilon\rangle \models \psi$, then $\mathcal{A}_{\mathcal{D},\psi}$ accepts $\langle T_{\mathcal{P}}, V_{\mathcal{P}}\rangle$. In fact, we show that there exists an accepting run $\langle T_r, r\rangle$ of $\mathcal{A}_{\mathcal{D},\psi}$ over $\langle T_{\mathcal{P}}, V_{\mathcal{P}}\rangle$. We define $\langle T_r, r\rangle$ as follows: the run starts at the initial state; thus $\epsilon \in T_r$ and $r(\epsilon) = (\epsilon, \psi)$. The run proceeds maintaining the invariant that for $y \in T_r$ with $r(y) = (x, \phi)$, we have $\langle T_{\mathcal{P}}, x\rangle \models \phi$. Since $\langle T_{\mathcal{P}}, \epsilon\rangle \models \psi$, the invariant holds for $y = \epsilon$. Also, by the semantics of CTLK and the definition of $\delta$, the run can always proceed such that all the successors $y \cdot (k, j)$ of a node $y$ that satisfies the invariant have $r(y \cdot (k, j)) = (x', \phi')$ with $\langle T_{\mathcal{P}}, x'\rangle \models \phi'$. Finally, the run always try to satisfy eventualities of $U$-formulae. Thus, whenever $\phi$ is of the form $A\phi_1 U\phi_2$ or $E\phi_1 U\phi_2$ and $\langle T_{\mathcal{P}}, x\rangle \models \phi_2$, it proceeds according to $\delta(\phi_2, V_{\mathcal{P}}(x), \boldsymbol{d}(x))$. It is easy to see that all the paths in such $\langle T_r, r\rangle$ are either finite or reach a state associated with a $\overline{U}$-formula or a $K_i$-formula and stay there thereafter. Thus, $\langle T_r, r\rangle$ is accepting. Finally, it is easy to check that the construction of the automaton $\mathcal{A}_{\mathcal{D},\psi}$ can be performed in linear time in the length of $\psi$. □

### 4.1 Example 2 – $\mathcal{L}_m$ to WAA

Here we demonstrate our technique by showing the translation of a $\mathcal{L}_m$ formula to a weak alternating tree automaton.

We take the formula $\varphi = AGK_ip$. To translate $\varphi$ into a WAA, we require the formula in a negation-normal form, plus with all abbreviations expanded; so we use $\varphi = A\left(false\overline{U}K_ip\right)$. The closure of $\varphi$ is $cl(\varphi) = \{\varphi, K_ip, p\}$. The elements of $cl(\varphi)$ define the states $Q$ of $\mathcal{A}_{\mathcal{D},\varphi}$. The accepting states $F$ are $\{\varphi, K_ip\}$. The alphabet of $\mathcal{A}_{\mathcal{D},\varphi}$ has only the letter $p$; transitions are therefore over $2^{\{p\}}$.

We formally define $\mathcal{A}_{\mathcal{D},\varphi} = \left(2^{\{p\}}, \mathcal{D}, \{\varphi, K_ip, p\}, \delta, \varphi, \{t, i\}, \{\varphi, K_ip\}\right)$; the transition relation $\delta$ is defined as follows:

| $q$ | $\delta(q,p,k)$ | $\delta(q,\emptyset,k)$ |
|---|---|---|
| $\varphi$ | $\bigwedge_{c=0}^{k_t-1}(c,t,\varphi) \wedge \bigwedge_{c=0}^{k_i-1}(c,i,p) \wedge \bigwedge_{c=0}^{k_i-1}(c,i,K_ip)$ | |
| $K_ip$ | $\bigwedge_{c=0}^{k_i-1}(c,i,p) \wedge \bigwedge_{c=0}^{k_i-1}(c,i,K_ip)$ | |
| $p$ | $\top$ | $\bot$ |

In the state $\varphi$ the automata expects that $\varphi$ recursively holds in all temporal successors and that both $p$ and $K_ip$ hold in all $i$-epistemically related states. This is highlighted in Rows 2 and 3 of the transition relation above.

The language of the product automaton obtained by composing $\mathcal{A}_{\mathcal{D},\varphi}$ and the tree unwinding of the IS from Figure 1, as specified in the following section, is non-empty.

## 5 The Product Automaton

Let $\mathcal{A}_{\mathcal{D},\psi} = \langle 2^P, \mathcal{D}, Q_\psi, \delta_\psi, q_0, A_t, F_\psi\rangle$ be an alternating tree automaton that accepts exactly all the $\mathcal{D}$-trees in $\mathcal{T}$ that satisfy $\psi$, as constructed in the previous section. Let $\mathcal{P} = \langle \mathcal{R}, s^0, I\rangle$ be an interpreted system such that the degrees of $\langle T_\mathcal{P}, V_\mathcal{P}\rangle$ are in $\mathcal{D}$. The product automaton of $\mathcal{A}_{\mathcal{D},\psi}$ and $\mathcal{P}$ is the alternating word automaton $\mathcal{A}_{\mathcal{P},\psi} = \langle\{a\}, \mathcal{S}\times Q_\psi, \delta, (s^0,q_0), F\rangle$ where $\delta$ and $F$ are defined as follows:

- Let $q \in Q_\psi$, $s \in \mathcal{S}$, $[s]_{\sim_i} = \langle s_{0,i}, \ldots, s_{d_i(s)-1,i}\rangle$ and $\{s' \in \mathcal{S} \mid$ there is $\rho \in \mathcal{R}, n \in \mathbb{N}$ such that $s = \rho(n)$ and $s' = \rho(n+1)\} = \langle s_{0,t}, \ldots, s_{d_t(s)-1,t}\rangle$. Further, let $\delta_\psi(q, I(s), \boldsymbol{d}(s)) = \theta$. Then $\delta((s,q),a) = \theta'$, where $\theta'$ is obtained from $\theta$ by replacing each atom $(c_i, j, q_i)$ in $\theta$ by the atom $(s_{c_i,j}, q_i)$.
- The acceptance condition $F$ is defined according to the acceptance condition $F_\psi$ of $\mathcal{A}_{\mathcal{D},\psi}$. If $F_\psi \subseteq Q_\psi$ is a Büchi condition, then $F = \mathcal{S}\times F_\psi$ is also a Büchi condition.

It is easy to see that $\mathcal{A}_{\mathcal{P},\psi}$ is of the same type as $\mathcal{A}_{\mathcal{D},\psi}$. In particular, if $\mathcal{A}_{\mathcal{D},\psi}$ is a $WAA$ (with a partition $\{Q_1,\ldots,Q_n\}$), then so is $\mathcal{A}_{\mathcal{P},\psi}$ (with a partition $\{\mathcal{S}\times Q_1, \ldots, \mathcal{S}\times Q_n\}$).

**Theorem 2.** *$\mathcal{L}(\mathcal{A}_{\mathcal{P},\psi})$ is nonempty iff $\psi$ is true in $\mathcal{P}$.*

**Proof.** We show that $\mathcal{L}(\mathcal{A}_{\mathcal{P},\psi})$ is nonempty if and only if $\mathcal{A}_{\mathcal{D},\psi}$ accepts the tree $\langle T_\mathcal{P}, V_\mathcal{P}\rangle$ built from the IS $\mathcal{P}$ as shown in Section 3. Since $\mathcal{A}_{\mathcal{D},\psi}$ accepts exactly all the $\mathcal{D}$-trees in $\mathcal{T}$ that satisfy $\psi$, and since all the degrees of $\mathcal{P}$ are in $\mathcal{D}$, the latter holds if and only if $\psi$ is true in $\mathcal{P}$. Given an accepting run of $\mathcal{A}_{\mathcal{D},\psi}$ over $\langle T_\mathcal{P}, V_\mathcal{P}\rangle$, we construct an accepting run of $\mathcal{A}_{\mathcal{P},\psi}$. Also, given an accepting run of $\mathcal{A}_{\mathcal{P},\psi}$, we construct an accepting run of $\mathcal{A}_{\mathcal{D},\psi}$ over $\langle T_\mathcal{P}, V_\mathcal{P}\rangle$.

Assume first that $\mathcal{A}_{\mathcal{D},\psi}$ accepts $\langle T_\mathcal{P}, V_\mathcal{P}\rangle$. Thus, there exists an accepting run $\langle T_r, r\rangle$ of $\mathcal{A}_{\mathcal{D},\psi}$ over $\langle T_\mathcal{P}, V_\mathcal{P}\rangle$. Recall that $T_r$ is labelled with $(\mathbb{N}\times A_t)^* \times Q_\psi$. A node $y \in T_r$ with $r(y) = (x,q)$ corresponds to a copy of $\mathcal{A}_{\mathcal{D},\psi}$ that is in the state $q$ and reads the tree obtained by unwinding $\mathcal{P}$ from $V_\mathcal{P}(x)$. Consider the

tree $\langle T_{r'}, r' \rangle$ where $T_{r'}$ is the tree obtained from $T_r$ by the function $f$ as follows. Suppose that $\delta_\psi(q, V_\mathcal{P}(x), \boldsymbol{d}(x)) = \theta$ and there are (possibly empty) sets $S_j = \{(c_0, j, q_0), \ldots, (c_{n_j}, j, q_{n_j})\} \subseteq \{0, \ldots, d_j(x) - 1\} \times \{j\} \times Q$ such that $\bigcup_{j \in A_t} S_j$ satisfies $\theta$, and for $0 \leq i < n_j$, we have $y \cdot (i, j) \in T_r$ and $r(y \cdot (i, j)) = (x \cdot (c_i, j), q_i)$. Then,

- $f(\epsilon) = \epsilon$;
- $f(y \cdot (i, j)) = f(y) \cdot (\Sigma_{j' < j}^{j \in A_t} n_{j'} + i)$.

The tree $T_{r'}$ is labelled with $0^* \times \mathcal{S} \times Q$, and for every $y \in T_r$ with $r(y) = (x, q)$, we have $r'(f(y)) = (0^{|x|}, V_\mathcal{P}(x), q)$. We show that $\langle T_{r'}, r' \rangle$ is an accepting run of $\mathcal{A}_{\mathcal{P},\psi}$. In fact, since $F = \mathcal{S} \times F_\psi$, we only need to show that $\langle T_{r'}, r' \rangle$ is a run of $\mathcal{A}_{\mathcal{P},\psi}$; acceptance follows from the fact that $\langle T_r, r \rangle$ is accepting. Consider a node $y \in T_r$ with $r(y) = (x, q)$, $V_\mathcal{P}(x) = s$, $[s]_{\sim_i} = \langle s_{0,i}, \ldots, s_{d_i(s)-1,i} \rangle$ and $\{s' \in \mathcal{S} \mid \text{there is } \rho \in \mathcal{R}, n \in \mathbb{N} \text{ such that } s = \rho(n) \text{ and } s' = \rho(n+1)\} = \langle s_{0,t}, \ldots, s_{d_t(s)-1,t} \rangle$. Further, we assumed that $\delta_\psi(q, I(s), \boldsymbol{k}) = \theta$, and since $\langle T_r, r \rangle$ is a run of $\mathcal{A}_{\mathcal{D},\psi}$, there exist sets $\{(c_0, j, q_0), \ldots, (c_n, j, q_n)\}$ satisfying $\theta$ such that the $j$-successors of $y$ in $T_r$ are $y \cdot (i, j)$ for $0 \leq i \leq n$, each labelled with $(x \cdot (c_i, j), q_i)$. In $\langle T_{r'}, r' \rangle$ by definition, $r'(f(y)) = (0^{|x|}, s, q)$ and the successors of $f(y)$ are $f(y \cdot (i, j))$, each labelled with $(0^{|x+1|}, s_{c_i,j}, q_i)$. Let $\delta((s, q), a) = \theta'$. By the definition of $\delta$, the sets $\{(s_{c_0,j}, q_0), \ldots, (s_{c_n,j}, q_n)\}$ satisfy $\theta'$. Thus, $\langle T_{r'}, r' \rangle$ is a run of $\mathcal{A}_{\mathcal{P},\psi}$.

Assume now that $\mathcal{A}_{\mathcal{P},\psi}$ accepts $a^\omega$. Thus, there exists an accepting run $\langle T_r, r \rangle$ of $\mathcal{A}_{\mathcal{P},\psi}$. Recall that $T_r$ is labelled with $0^* \times \mathcal{S} \times Q_\psi$. Consider the tree $\langle T_{r'}, r' \rangle$ labelled with $(\mathbb{N} \times A_t)^* \times Q_\psi$, where $T_{r'}$ and $r'$ are obtained from $T_r$ and $r$ by means of a function $g : T_r \to T_{r'}$ as follows:

- $g(\epsilon) = \epsilon$ and $r'(\epsilon) = (\epsilon, q_0)$;
- if $y \cdot c \in T_r$, $r'(g(y)) \in \{x\} \times Q_\psi$, $r(y \cdot c) = (0^{|x+1|}, s, q)$ and $i$, $j$ are such that $V_\mathcal{P}(x \cdot (i, j)) = s$, then $g(y \cdot c) = g(y) \cdot (i, j)$ and $r'(g(y \cdot c)) = (x \cdot (i, j), q)$.

As in the previous direction, we can check that $\langle T_{r'}, r' \rangle$ is an accepting run of $\mathcal{A}_{\mathcal{D},\psi}$ over $\langle T_\mathcal{P}, V_\mathcal{P} \rangle$. □

## 6 Conclusions

In this paper we presented a translation between the branching time temporal-epistemic logic CTLK and alternating automata over infinite trees. We have shown that the language accepted by an automata representing the composition of a temporal-epistemic formula and an interpreted system is non-empty if and only if the interpreted system satisfies the formula. We illustrated this technique through an example.

Automata-theoretic approaches for knowledge are only the first step towards a multitude of other techniques for epistemic knowledge. For example, the implementation of partial order reduction techniques for temporal logic often relies on automata-theoretic techniques, while the application of partial order reduction in agent systems currently only considers the hypothetical reduction

in the state space [4]. Providing an automata-theoretic approach to agent-based logics allows for the possible implementation of these techniques. Alternating automata over infinite trees, in a temporal-epistemic context, may also allow for truly on-the-fly verification of multi-agent systems.

The automata construction of temporal logic formulae allows for reasoning about problems such as implication and satisfiability of formulae [3]. An automata construction for temporal-epistemic logic may also lead to a solution to these problems in the agent realm.

Our initial further work will be focused on constructing an implementation of these techniques, including a comparison to the existing symbolic approaches to the same problem. While literature exists documenting several implementations of automata theoretic branching time model checkers [12,10], there appears to be infrequent adoption of such implementations. This is possibly due to inefficiencies or unavailability of tools. We believe the space complexity of the techniques presented here will correspond closely to the existing space complexity bounds of the temporal-only case [3]. The time complexity bounds may have to be reconsidered, as additional steps are required to match epistemic successors.

## References


1. O. Bernholtz and O. Grumberg. Branching Time Temporal Logic and Amorphous Tree Automata. In *Proc. of CONCUR '93*, LNCS, pages 262–277. Springer, 1993.
2. R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning About Knowledge*. MIT Press, 1995.
3. O. Kupferman, M. Y. Vardi, and P. Wolper. An Automata-Theoretic Approach to Branching-Time Model Checking. *J. ACM*, 47(2):312–360, 2000.
4. A. Lomuscio, W. Penczek, and H. Qu. Partial Order Reduction for Model Checking Interleaved Multi-Agent Systems. *Fund. Informaticae*, 2010. To appear.
5. A. Lomuscio, H. Qu, and F. Raimondi. MCMAS: A Model Checker for the Verification of Multi-Agent Systems. In *Proceedings of CAV '09*, volume 5643 of *LNCS*, pages 682–688. Springer, 2009.
6. J.-J. C. Meyer and W. Hoek. *Epistemic Logic for AI and Computer Science*. Cambridge University Press, 1995.
7. D. Muller, A. Saoudi, and P. Schupp. Alternating Automata. The Weak Monadic Theory of the Tree, and its Complexity. In *ICALP*, volume 226 of *LNCS*, pages 275–283. Springer, 1986.
8. D. Muller and P. Schupp. Alternating Automata on Infinite Trees. *Theoretical Computer Science*, 54:267–276, 1987.
9. W. Penczek and A. Lomuscio. Verifying Epistemic Properties of Multi-Agent Systems via Bounded Model Checking. In *Proceedings of AAMAS 03*, pages 209–216. ACM, 2003.
10. K. Qian and A. Nymeyer. Language-Emptiness Checking of Alternating Tree Automata Using Symbolic Reachability Analysis. *ETCS*, 149(2):33–49, 2006.
11. R. van der Meyden and N. V. Shilov. Model Checking Knowledge and Time in Systems with Perfect Recall (Extended Abstract). In *Proceedings of FSTTCS '99*, volume 1738 of *LNCS*, pages 432–445. Springer, 1999.
12. W. Visser. *Efficient CTL* Model Checking using Games and Automata*. PhD thesis, University of Manchester, UK, 1998.