The Complexity of Model Checking Knowledge and Time

Laura Bozzelli
Università degli Studi di Napoli "Federico II"

Bastien Maubert
Università degli Studi di Napoli "Federico II"

Aniello Murano
Università degli Studi di Napoli "Federico II"

Abstract

We establish the precise complexity of the model checking problem for the main logics of knowledge and time with perfect recall. While it was known to be PSPACE-complete for extensions of LTL, CTL and CTL* with knowledge in the case of memoryless agents, for agents with perfect recall the problem was only known to be nonelementary, with a number of exponentials that increases with the nesting of knowledge operators in the formula. The precise complexity of the problem when the maximum nesting is fixed has been an open problem for twenty years. We close it by establishing improved upper bounds for CTL* with knowledge, which also hold for the fragments CTL and LTL with knowledge, and providing matching lower bounds for all three logics. Moreover, we do so for the two main variants of perfect recall, namely synchronous and asynchronous perfect recall, thus essentially closing the picture for the complexity of model checking logics of knowledge and time.

2012 ACM Subject Classification Theory of computation → Complexity theory and logic; Theory of computation → Logic and verification; Theory of computation → Modal and temporal logics; Computing methodologies → Reasoning about belief and knowledge

Keywords and phrases Model checking, epistemic temporal logic, complexity

1 Introduction

A central aspect of multi-agent systems is that agents only have partial knowledge about the system [21]. Epistemic logics are a standard framework to reason about what agents know about the world and each others' knowledge. In order to talk about behaviours of on-going multi-agent systems, epistemic logics have been combined with temporal logics such as LTL [22], CTL and CTL* [8]. The resulting *epistemic temporal logics* can express properties of the evolution of agents' knowledge over time. These logics have been applied to the modelling and analysis of, e.g., distributed protocols [19, 11], information flow and cryptographic protocols [28, 12] and knowledge-based programs [29].

The satisfiability problem for this family of logics has been thoroughly studied in [14, 15], which categorise epistemic temporal logics according to a number of criteria: (1) is the system synchronous or asynchronous; (2) does it have a unique initial state known to all agents; (3) do agents have bounded memory or perfect recall; (4) can agents learn; (5) is the temporal part of the language linear or branching; (6) can the epistemic part of the language talk about the knowledge of several agents; and (7) can it talk about common knowledge. By considering all the possible combinations, the authors identify 96 logics and study their satisfiability/validity problem. Sound and complete axiomatization for those of these logics that admit one are also provided in [13].

While the picture is clear for satisfiability and axiomatization of these logics, it is not entirely the case for the model-checking problem, which is arguably at least as important for the verification of multi-agent systems as satisfiability or axiomatization (see for instance [6]).

	LTL	CTL	CTL*
bm, asyn, K/CK	PSPACE-c	PTIME-c	PSPACE-c
bm, syn, K/CK	PSPACE-c	PSPACE-c	PSPACE-c
pr, syn/asyn, CK	undecidable	undecidable	undecidable
pr, syn/asyn, K	$(k-1)$-EXPSPACE-c	$(k-1)$-EXPSPACE-c	$(k-1)$-EXPSPACE-c

Table 1 Known and new results (in grey; for CTL, the upper bounds were known). "bm" and "pr" stand for "bounded memory" and "perfect recall", "syn" and "asyn" for "synchronous" and "asynchronous", and "CK" indicates extensions with both knowledge and common knowledge operators, while "K" indicates the absence of common knowledge. Finally $k \geq 1$ is the maximal alternation depth of formulas.

For agents with bounded memory the situation is well understood. In particular, for the asynchronous setting, adding knowledge operators and even common-knowledge operators to LTL, CTL or CTL* does not increase the complexity of model checking: it is PSPACE-complete for extensions of LTL and CTL* [18] and PTIME-complete for extensions of CTL [24]. For the synchronous setting, the situation is similar, but for extensions of CTL, the problem becomes PSPACE-complete [10, 17].

For agents with perfect recall, the problem is undecidable when common knowledge is part of the language [27]. For the extensions of LTL, CTL and CTL* with knowledge but no common knowledge operators, denoted respectively LTLK, CTLK and CTLK*, model checking is instead decidable but non-elementary [27, 1, 7, 2, 4]. It was noted that the non-elementary blow-up depends on the *alternation depth* of formulas, the maximal number of alternations between knowledge operators for different agents: each additional alternation forces to maintain in the model-checking procedure an additional layer of information about what agents know. For a fixed alternation depth $k \geq 1$, in the synchronous setting model checking is known to be in k-EXPSPACE for LTLK [27]. For both the synchronous and asynchronous semantics, it is known to be in $(k-1)$-EXPSPACE for CTLK [1], and in k-EXPTIME for CTLK* [2, 4]. However it is not known whether these bounds are tight.

We show that they are tight only for CTLK: we prove that model-checking for LTLK, CTLK and CTLK* is actually $(k-1)$-EXPSPACE-complete for alternation depth at most k, both for synchronous and asynchronous semantics. The upper bounds for synchronous and asynchronous CTLK* and LTLK are new, and the lower bounds are new for all six logics. We summarise the main results in Table 1. We point out that our PSPACE-completeness result for the fragment of LTLK with alternation depth one generalises that for synchronous LTLK with one agent proved in [10].

Note that the complexity of model checking is often studied for models that are given by an explicit description of their states and transitions, which is what we consider in this work. More recently some works started to study the complexity of model checking multi-agent systems for succinct representations [20, 16].

2 Preliminaries

Let $\mathbb{N}$ be the set of natural numbers. For all $n, k \in \mathbb{N}$, $\mathit{Tower}(n, k)$ denotes a tower of exponentials of height k and argument n: define $\mathit{Tower}(n, 0) = n$ and $\mathit{Tower}(n, k+1) = 2^{\mathit{Tower}(n,k)}$. Let k-EXPSPACE be the class of languages decided by deterministic Turing machines bounded in space by functions of n in $O(\mathit{Tower}(n^c, k))$, for some constant $c \geq 1$.

Let w be a finite or infinite word over some finite alphabet Σ. We denote by $|w|$ the length of w (we set $|w| = \infty$ if w is infinite). For all $0 \leq i, j < |w|$, with $i \leq j$, w_i is the i-th letter of w, $w_{\leq i}$ is the prefix of w that ends at position i, $w_{\geq i}$ is the suffix that starts at position i, and $w_{[i,j]} = w_i \dots w_j$. For words w and w', we write $w \preccurlyeq w'$ if w is a prefix of w'.

2.1 Epistemic Temporal Logics

We recall the logic CTLK^* and its fragments LTLK and CTLK, which respectively correspond to extensions of CTL^*, LTL and CTL with knowledge operators.

Let us fix a countably infinite set of atomic propositions $\mathcal{AP}$ and a finite set of agents Ag. As for state and path formulas in CTL^*, we distinguish between *history formulas* and *path formulas.* We say history formulas instead of state formulas because, considering agents with *perfect recall* of the past, the truth of epistemic formulas depends not only on the current state, but also on the history before reaching this state.

▶ **Definition 1** (Syntax of CTLK^*)**.** *The sets of history formulas φ and path formulas ψ are defined by the following grammar:*

$$\begin{array}{lll} \varphi & ::= & p \mid \neg\varphi \mid \varphi \vee \varphi \mid E\psi \mid K_a\varphi \\ \psi & ::= & \varphi \mid \neg\psi \mid \psi \vee \psi \mid X\psi \mid \psi U \psi, \end{array}$$

where $p \in \mathcal{AP}$ and $a \in Ag$.

Operators X and U are the standard *next* and *until* temporal operators of LTL, E is the existential path quantifier of CTL^*, and K_a is the knowledge operator for agent a from epistemic logics. Formula $K_a\varphi$ reads as "agent a knows that φ is true". As usual we define $A\psi := \neg E \neg\psi$. The language of CTLK^* consists of the history formulas. We let $\mathrm{Sub}(\varphi)$ be the set of subformulas in φ, and we define the *size* of a formula φ as $|\varphi| = |\mathrm{Sub}(\varphi)|$. We call *alternation depth* of a formula φ, written $\mathrm{ad}(\varphi)$, the maximum number of alternations between knowledge operators for different agents in the formula. For instance, $\mathrm{ad}(p) = 0$, $\mathrm{ad}(K_a p) = 1$, $\mathrm{ad}(K_a \neg K_a p) = 1$ and $\mathrm{ad}(K_b K_a q \vee K_a p) = 2$.

Fragments of CTLK^* We consider two usual syntactic fragments of CTLK^*, namely LTLK and CTLK. LTLK consists of the formulas of the form $A\psi$ or $E\psi$ where every path operator A in ψ is immediately preceded by a knowledge operator K_a. CTLK is obtained by requiring that the temporal modalities X and U are immediately preceded by a path quantifier. For every $k \in \mathbb{N}$ we define LTLK_k, CTLK_k and CTLK^*_k the fragments of LTLK, CTLK and CTLK^*, respectively, obtained by restricting to formulas of alternation depth at most k.

Semantics CTLK^* formulas are interpreted over Kripke structures equipped with one *indistinguishability relation* $\sim_a$ for each agent a.

▶ **Definition 2.** *A* Kripke structure *(KS) is a structure $M = (\mathrm{AP}, S, R, V, \{\sim_a\}_{a \in Ag}, s^\iota)$, where $\mathrm{AP} \subset \mathcal{AP}$ is a finite subset of atomic propositions, S is a set of states, $R \subseteq S \times S$ is a left-total*[1] *transition relation, $V : S \to 2^{\mathrm{AP}}$ is a valuation function, $\sim_a \subseteq S \times S$ is an equivalence relation, for each $a \in Ag$, and $s^\iota \subseteq S$ is an initial state.*

The *size* $|M|$ of M is the number of states in M. A *path* is an infinite sequence of states $\pi = s_0 s_1 \dots$ such that for all $i \geq 0$, $s_i R s_{i+1}$, and a *history* τ is a non-empty prefix of a path. We denote by $\mathrm{Hist}(s)$ (resp. $\mathrm{Path}(s)$) the set of histories (resp. paths) that start in s. Unless specified otherwise, all histories and paths are assumed to start in the initial state s^ι. For

[1] i.e., for every $s \in S$ there exists $s' \in S$ such that sRs'

$I \subseteq S$, we write $R(I) = \{s' \mid \exists s \in I \text{ s.t. } sRs'\}$ for the set of successors of states in I. Finally, for $a \in Ag$ and $s \in S$, we let $[s]_a$ be the equivalence class of s for relation $\sim_a$.

We consider the classic *synchronous* and *asynchronous perfect recall semantics* of knowledge modalities, where agents remember all of the past. While in synchronous systems, agents always observe when a transition takes place, in asynchronous ones agents cannot tell that a transition occurred if their observation of the state remains unchanged.

▶ **Definition 3.** *Two histories τ and τ' are indistinguishable for an agent a with* synchronous perfect recall *(SPR for short), written $\tau \approx^s_a \tau'$, if they are point-wise indistinguishable to a, i.e. $|\tau| = |\tau'|$ and $\tau_i \approx_a \tau'_i$ for each $i < |\tau|$.*

To define asynchronous perfect recall, we first define the sequence of observations that an agent has along a history, in which sequences of successive identical observations collapse to a single observation. Formally, for an agent a, we let $\mathrm{Obs}_a(s) = [s]_a$, $\mathrm{Obs}_a(\tau \cdot s) = \mathrm{Obs}_a(\tau) \cdot [s]_a$ if $[s]_a \neq \mathrm{lst}(\mathrm{Obs}_a(\tau))$, and $\mathrm{Obs}_a(\tau)$ otherwise.

▶ **Definition 4.** *Two histories τ and τ' are indistinguishable for an agent a with* asynchronous perfect recall *(APR for short), written $\tau \approx^{as}_a \tau'$, if $Obs_a(\tau) = Obs_a(\tau')$.*

▶ **Definition 5** (Semantics). *Fix a model M. A history formula φ is evaluated on a history τ. A path formula ψ is interpreted on a path π and a point in time $n \in \mathbb{N}$. The SPR (resp., APR) semantics is defined by induction as follows:*

$$
\begin{array}{lll}
\tau \models p & \textit{if} & p \in V(\mathrm{lst}(\tau)) \\
\tau \models \neg\varphi & \textit{if} & \tau \not\models \varphi \\
\tau \models \varphi_1 \vee \varphi_2 & \textit{if} & \tau \models \varphi_1 \textit{ or } \tau \models \varphi_2 \\
\tau \models E\psi & \textit{if} & \exists \pi \textit{ s.t. } \tau \preccurlyeq \pi \textit{ and } \pi, |\tau| - 1 \models \psi \\
\tau \models K_a\varphi & \textit{if} & \forall \tau' \in \mathit{Hist}(s^\iota) \textit{ such that } \tau' \approx^s_a \tau \,(\textit{resp., } \tau' \approx^{as}_a \tau),\ \tau' \models \varphi \\
\pi, n \models \varphi & \textit{if} & \pi_{\leq n} \models \varphi \\
\pi, n \models \neg\psi & \textit{if} & \pi, n \not\models \psi \\
\pi, n \models \psi_1 \vee \psi_2 & \textit{if} & \pi, n \models \psi_1 \textit{ or } \pi, n \models \psi_2 \\
\pi, n \models X\psi & \textit{if} & \pi, (n+1) \models \psi \\
\pi, n \models \psi_1 U \psi_2 & \textit{if} & \exists m \geq n \textit{ s.t. } \pi, m \models \psi_2 \textit{ and } \forall k \textit{ s.t. } n \leq k < m,\ \pi, k \models \psi_1
\end{array}
$$

A model M with initial state s^ι *satisfies* a CTLK* formula φ under the SPR (resp., APR) semantics, written $M \models_{\mathrm{sy}} \varphi$ (resp., $M \models_{\mathrm{as}} \varphi$), if $s^\iota \models \varphi$ under the SPR (resp., APR) semantics.

▶ Remark 6. Observe that we assume the definition of the model to be common knowledge among the agents, which is standard for instance in game theory. As a result the initial state is common knowledge, which corresponds to the *unique initial state* assumption in [15, 13]. This is reflected in the semantics in the fact that we only consider indistinguishable histories that start in the initial state. We note that one can simulate in this semantics the case where the initial state is unknown by adding an artificial initial state $s^\iota{}_0$ such that $(s^\iota{}_0, s) \in R$ and $(s, s^\iota{}_0) \notin R$ for all states s in the original model, and labelling each state s with an atom p_s. Then evaluating $s \models \varphi$ (with unknown initial state) is equivalent to evaluating $s^\iota{}_0 \models AX(p_s \to \varphi)$ (with unique initial state).

2.2 Main result

The *model-checking problem* for the SPR (resp., APR) semantics, of which we study the complexity, is checking for a finite model M and a CTLK* formula φ, whether $M \models_{\mathrm{sy}} \varphi$ (resp., $M \models_{\mathrm{as}} \varphi$). In this work we prove the following result.

▶ **Theorem 7.** *For every $k \in \mathbb{N}$, the model-checking problem for LTLK_{k+1}, CTLK_{k+1} and CTLK^*_{k+1} is k-*EXPSPACE*-complete, both for SPR and APR semantics.*

3 Powerset construction

In this section we recall a classic powerset construction for algorithmic questions related to imperfect information. It was first used by Reif in [25] to eliminate imperfect information from two-player games, and in [26, 27, 1] to model check variants of LTLK and CTLK. We show that it also can be used to model-check CTLK*.

While the powerset construction in [26, 27, 1] makes use of k-trees, which encore enough information to evaluate formulas of alternation depth k, we instead consider the simpler case of alternation 1. We first show how to model check formulas of alternation depth one in polynomial space by constructing 1-trees on the fly. We next obtain our upper bounds for the general case by using the powerset construction to reduce the model-checking problem for formulas of alternation depth $k+1$ to that of alternation depth k, with an exponential blowup in the model. The only difference between the synchronous and asynchronous case will be in the definition of the 1-trees, which are essentially information sets.

3.1 Information sets and updates

Since we only need the degenerate case of 1-trees, which are a collection of *information sets* together with the current state of the system, we only define information sets and not 1-trees. The notion of information set is meant to capture the set of states that an agent considers possible at a given moment, and the following definition is common to synchronous and asynchronous perfect recall.

▶ **Definition 8** (Information sets). *Given a model M with state set S, an* information set *$I \subseteq S$ is a set of states. Given a history τ and an agent a, the* information set of a at τ *is defined as*

$$I_a(\tau) = \{s \mid \exists \tau' \in \mathit{Hist}(s^\iota) \text{ s.t. } \tau \approx_a \tau' \text{ and } s = \mathrm{lst}(\tau')\}.$$

We will write I_a^{sy} when referring to the synchronous semantics, and I_a^{as} for the asynchronous one, and we may omit the subscript when clear from the context.

We now define two different update functions, for the synchronous and asynchronous cases. The role of these functions is to compute the new information set of an agent after a transition, given her former information set and the new state. We start with the synchronous case, which is easier and standard (see for instance [25, 5, 23], or [26, 27] for the more general case of k-trees).

▶ **Definition 9** (Synchronous update). *The* synchronous update *of an information set I_a for agent a with a new state s is*

$$\mathrm{Up}^{\mathrm{sy}}(I_a^{\mathrm{sy}}, s) = R(I_a^{\mathrm{sy}}) \cap [s]_a$$

This definition says that the set of states that agent a considers possible after taking a transition that arrives in state s consists of all states that are successors of states she previously considered possible, and that are compatible with what she observes of the new state. The following fact is folklore and follows directly by applying the definitions:

▶ **Lemma 10.** *For every history $\tau' = \tau \cdot s$ and agent a, $I_a^{\mathrm{sy}}(\tau') = \mathrm{Up}^{\mathrm{sy}}(I_a^{\mathrm{sy}}(\tau), s)$.*

For asynchronous perfect recall, the update is slightly more involved, as the agent may consider that arbitrarily many steps occurred that did not change his observation. We call *invisible step* (for some agent a) a transition between two states s and s' such that $s \sim_a s'$, and given a set of states S, we let $\mathrm{Reach}_i^a(S) \supseteq S$ be the set of states reachable from S via steps invisible for a. We can now define the update as follows:

▶ **Definition 11** (Asynchronous update). *The* asynchronous update *of an information set I_a for agent a with a new state s is*

$$\mathrm{Up}^{\mathrm{as}}(I_a^{\mathrm{as}}, s) = \begin{cases} I_a^{\mathrm{as}} & \text{if } I_a^{\mathrm{as}} \subseteq [s]_a, \\ \mathit{Reach}_i^a(R(I_a^{\mathrm{as}}) \cap [s]_a) & \text{otherwise.} \end{cases}$$

This definition is an adaptation to our setting of the one in [23], which considers two-player games. The case of k-trees for asynchronous perfect recall is considered in [26]. We have the following result, corresponding to Lemma 10, and which also follows from the definitions:

▶ **Lemma 12.** *For every history $\tau' = \tau \cdot s$ and agent a, $I_a^{\mathrm{as}}(\tau') = \mathrm{Up}^{\mathrm{as}}(I_a^{\mathrm{as}}(\tau), s)$.*

3.2 Powerset construction

We now define the powerset construction which transforms a model M into another model $\widehat{M}$ of exponential size in which epistemic formulas of alternation depth 1 can be evaluated positionally. The idea is that states of $\widehat{M}$ contain, in addition to the current state, the current information set of each agent (thus positions are 1-trees in the sense of [27]). This construction can be instantiated either for the synchronous or asynchronous semantics by choosing the appropriate update function for information sets. In the rest of this section we will often omit to specify in the notations which case is considered, because the reasoning will work for both semantics. The only difference is when we need the correctness of the update functions: in one case we will refer to Lemma 10, and in the other to Lemma 12.

▶ **Definition 13** (Powerset construction). *Given $M = (\mathrm{AP}, S, R, V, \{\sim_a\}_{a\in Ag}, s^\iota)$, we define $\widehat{M} = (\mathrm{AP}, \widehat{S}, \widehat{R}, \widehat{V}, \{\widehat{\sim}_a\}_{a\in Ag}, \widehat{s}^\iota)$, where*

- $\widehat{S} = S \times (2^S)^{Ag}$
- $(s, \langle I_a\rangle_{a\in Ag})\widehat{R}(s', \langle I'_a\rangle_{a\in Ag})$ *if sRs' and for each $a \in Ag$, $I'_a = \mathrm{Up}(I_a, s')$*
- $\widehat{V}(s, \langle I_a\rangle_{a\in Ag}) = V(s)$
- *for each $b \in Ag$, $(s, \langle I_a\rangle_{a\in Ag})\widehat{\sim}_b(s', \langle I'_a\rangle_{a\in Ag})$ if $s' \in I_b$ and $I_b = I'_b$*
- $\widehat{s}^\iota = (s^\iota, \langle\{s^\iota\}\rangle_{a\in Ag})$

Observe that because the update of information sets with a new state is deterministic, every history τ in M defines a unique history $\widehat{\tau}$ of length $|\tau|$ in $\widehat{M}$, that starts in $\widehat{s}^\iota$ and follows transitions determined by τ. Formally, $\widehat{\tau}$ is defined by induction as follows: $\widehat{s^\iota} = \widehat{s}^\iota$, and $\widehat{\tau \cdot s} = \widehat{\tau} \cdot (s, \langle I'_a\rangle_{a\in Ag})$, where $(s, \langle I'_a\rangle_{a\in Ag})$ is the unique successor of $\mathrm{lst}(\widehat{\tau})$ in $\widehat{M}$ whose first component is s. Similarly we let $\widehat{\pi}$ be the infinite path induced by π in $\widehat{M}$. The following lemma follows directly by definition of $\widehat{M}$, $\widehat{\tau}$ and application of Lemma 10 or Lemma 12, depending on the semantics considered:

▶ **Lemma 14.** *For every history τ, it holds that $\mathrm{lst}(\widehat{\tau}) = (\mathrm{lst}(\tau), \langle I_a(\tau)\rangle_{a\in Ag})$.*

Observe that $|\widehat{M}| \leq |M|2^{|Ag||M|}$ (it may be smaller because we can remove states that are not reachable by the transition relation and the indistinguishability relations).

We now describe how formulas of alternation depth one can be evaluated positionally in the powerset model $\widehat{M}$.

▶ **Definition 15** (Alternative semantics). *Given a powerset model $\widehat{M}$, a state $\widehat{s} \in \widehat{S}$ and a path $\widehat{\pi}$, the alternative semantics of a formula of alternation depth at most 1 is defined inductively as follows:*

$$
\begin{array}{lll}
\widehat{s} \models_I p & \textit{if} & p \in \widehat{V}(\widehat{s}) \\
\widehat{s} \models_I \neg\varphi & \textit{if} & \widehat{s} \not\models_I \varphi \\
\widehat{s} \models_I \varphi_1 \vee \varphi_2 & \textit{if} & \widehat{s} \models_I \varphi_1 \textit{ or } \widehat{s} \models_I \varphi_2 \\
\widehat{s} \models_I A\psi & \textit{if} & \textit{for all } \widehat{\pi} \in \textit{Path}(\widehat{s}),\ \ \widehat{\pi} \models_I \psi \\
\widehat{s} \models_I K_a\varphi & \textit{if} & \textit{for all } \widehat{s}' \textit{ s.t. } \widehat{s}' \mathrel{\widehat{\sim}_a} \widehat{s},\ \ \widehat{s}' \models_I \varphi \\
\widehat{\pi} \models_I \varphi & \textit{if} & \widehat{\pi}_0 \models_I \varphi \\
\widehat{\pi} \models_I \neg\psi & \textit{if} & \widehat{\pi} \not\models_I \psi \\
\widehat{\pi} \models_I \psi_1 \vee \psi_2 & \textit{if} & \widehat{\pi} \models_I \psi_1 \textit{ or } \widehat{\pi} \models_I \psi_2 \\
\widehat{\pi} \models_I X\psi & \textit{if} & \widehat{\pi}_{\geq 1} \models_I \psi \\
\widehat{\pi} \models_I \psi_1 U \psi_2 & \textit{if} & \exists i \geq 0 \textit{ such that } \widehat{\pi}_{\geq i} \models_I \psi_2 \textit{ and } \forall j \textit{ such that } 0 \leq j < i,\ \widehat{\pi}_{\geq j} \models_I \psi_1
\end{array}
$$

Note that we cannot evaluate nested knowledge operators for different agents because we have only one "level" of knowledge in the states of $\widehat{M}$. However we can evaluate nested knowledge operators for a same agent because indistinguishability relations between states $\sim_a$, and thus also relations $\approx_a$ between histories, are equivalence relations: if $h \approx_a h'$, then agent a knows the same things in h and in h'. This is reflected in the definition of the indistinguishability relations in $\widehat{M}$.

The following proposition establishes that this alternative semantics is equivalent to the original one for formulas of alternation depth at most one.

▶ **Proposition 16.** *For every history formula φ and path formula ψ of alternation depth at most one, each model M, history τ, path π and time $n \in \mathbb{N}$,*

$$
\begin{array}{rll}
\tau \models \varphi & \textit{iff} & \mathrm{lst}(\widehat{\tau}) \models_I \varphi, \textit{ and} \\
\pi, n \models \psi & \textit{iff} & \widehat{\pi}_{\geq n} \models_I \psi.
\end{array}
$$

Proof. The proof is by induction on formulas. We only treat the cases of atomic propositions, path quantifier and knowledge operators, all remaining cases follow directly by definition of the semantics and application of the induction hypothesis.

$\varphi = p$: By Lemma 14, $\widehat{\tau}$ ends in state $(\mathrm{lst}(h), \langle I_a(\tau)\rangle_{a \in Ag})$, so by definition of $\widehat{M}$ we have $\widehat{V}(\mathrm{lst}(\widehat{\tau})) = V(\mathrm{lst}(\tau))$, and the result follows.

$\varphi = A\psi$: It is enough to observe that $\mathrm{Path}(\mathrm{lst}(\widehat{\tau})) = \{\widehat{\pi}_{\geq |\tau|-1} \mid \tau \preccurlyeq \pi\}$; the result then follows by induction hypothesis.

$\varphi = K_a\varphi'$: Let us write $\mathrm{lst}(\widehat{\tau}) = (s, \langle I_a\rangle_{a \in Ag})$, and recall that by Lemma 14,

$$I_a = I_a(\tau) = \{\mathrm{lst}(\tau') \mid \tau \approx_a \tau'\}. \tag{1}$$

For the first direction assume that $\mathrm{lst}(\widehat{\tau}) \models_I K_a\varphi'$. Now let $\tau' \approx_a \tau$, we show that $\tau' \models \varphi'$, and we are done. Since $\mathrm{lst}(\widehat{\tau}) \models_I K_a\varphi'$ and, by (1), $\mathrm{lst}(\tau') \in I_a$, we have by definition of $\widehat{M}$ and $\models_I$ that $(\mathrm{lst}(\tau'), \langle I'_a\rangle_{a \in Ag}) \models_I \varphi'$, where $I'_a = I_a = I_a(\tau) = I_a(\tau')$. Now, since $K_a\varphi'$ is of alternation depth one, φ' does not contain any operator K_b for $b \neq a$. As a result the semantics of φ' does not depend on I_b for $b \neq a$, and it is also the case that $(\mathrm{lst}(\tau'), \langle I_a(\tau')\rangle_{a \in Ag}) \models_I \varphi'$. By induction hypothesis we conclude that $\tau' \models \varphi'$.

For the other direction assume that $\tau \models K_a\varphi'$. To show that $\mathrm{lst}(\widehat{\tau}) \models_I K_a\varphi'$, we take some $\widehat{s}' \mathrel{\widehat{\sim}_a} \mathrm{lst}(\widehat{\tau})$ and show that $\widehat{s}' \models_I \varphi'$. By definition of $\widehat{M}$, $\widehat{s}'$ is of the form

$\widehat{s}' = (s', \langle I'_a \rangle_{a \in Ag})$ with $s' \in I_a(\tau)$ and $I'_a = I_a(\tau)$. Let τ' be such that $\tau' \approx_a \tau$ and $\text{lst}(\tau') = s'$. Since $\tau \models K_a\varphi'$, we have $\tau' \models \varphi'$, and by induction hypothesis $\text{lst}(\widehat{\tau'}) \models_I \varphi'$. Now, by Lemma 14, $\text{lst}(\widehat{\tau'}) = (s', \langle I_a(\tau') \rangle_{a \in Ag})$, and since $\tau \approx_a \tau'$ we have $I_a(\tau) = I_a(\tau')$. Again, because $K_a\varphi'$ has alternation depth one, φ' contains no K_b for $b \neq a$, and thus $\widehat{s}' \models_I \varphi'$.

■

4 Upper bounds

In this section we establish the upper bounds in Theorem 7. We start with the base case, showing that we can model check formulas of alternation depth at most one in polynomial space. We then use a marking algorithm on the powerset construction to show that model-checking formulas of alternation depth $k+1$ reduces to model-checking formulas of alternation depth k on an exponentially larger model. All the reasoning in this section is independent of the chosen semantics, synchronous or asynchronous perfect recall.

4.1 Alternation depth one

In the main model-checking procedure, formulas of the form $E\psi$ are dealt with by guessing a path in the powerset model, that is built on the fly, and evaluating ψ on it in polynomial space. However for our algorithm to terminate, we need to bound the length of paths that need to be searched, and we need this bound to be at most exponential so that we can count up to it in polynomial space. We now prove that it is indeed the case.

An infinite word w is *ultimately periodic* if there exist $i, j \in \mathbb{N}$ such that $w = w_{\leq i-1} w^*_{[i,j]}$. Letting i and j be the smallest such values, we call i the *start index* of π, and $j - i + 1$ is called its *period*.

▶ **Lemma 17.** *Let ψ be a* CTLK* *path formula of alternation depth at most 1, let M be a model, $\widehat{M}$ the powerset model, and $\widehat{s}$ a state in $\widehat{M}$. If $\widehat{s} \models E\psi$, then there exists a path $\widehat{\pi}$ starting in $\widehat{s}$ such that $\widehat{\pi} \models_I \psi$ and $\widehat{\pi}$ is ultimately periodic with start index and period less than $|M|2^{|Ag||M|+|\psi|}$.*

Proof. As in the proof of Proposition 19, we can evaluate maximal history subformulas of ψ in states of $\widehat{M}$, marking it with fresh atoms AP_f and replacing maximal history subformulas in ψ with these atoms. We thus get a marked powerset model $\widehat{M}'$ and a formula ψ' of alternation depth 0 such that $\widehat{\pi} \models_I \psi$ iff $\widehat{\pi}' \models_I \psi'$.

Since ψ' is an LTL formula, one can build a nondeterministic Büchi word automaton $\mathcal{A}_{\psi'}$ of size at most $2^{|\psi'|}$ that accepts precisely the infinite words on $2^{\text{AP} \cup \text{AP}_f}$ that satisfy ψ' [30]. By taking the product of $\mathcal{A}_\psi$ with $\widehat{M}'$, we obtain an automaton $\mathcal{A}_{\widehat{M}',\psi'}$ over the states of $\widehat{M}'$ that has size at most $|M|2^{|Ag||M|+|\psi'|}$ and accepts precisely paths in $\widehat{M}'$ that satisfy ψ'. By definition of the Büchi acceptance condition, there exists such a path if and only if there is an ultimately periodic path in the transition graph of $\mathcal{A}_{\widehat{M}',\psi'}$ with an accepting state in the periodic part. So if there is such a path $\widehat{\pi}'$, there is an ultimately periodic one of start index and period less than the size of $\mathcal{A}_{\widehat{M}',\psi'}$, i.e. $|M|2^{|Ag||M|+|\psi|}$. Finally, observe that this path $\widehat{\pi}'$ such that $\widehat{\pi}' \models_I \psi'$ defines an ultimately periodic path $\widehat{\pi}$ in the unmarked powerset model $\widehat{M}$, with same start index and period, such that $\widehat{\pi} \models_I \psi$. ■

We now adapt Emerson and Lei's algorithm, which shows how to turn a polynomial-space model-checking procedure for LTL into one for CTL* that also runs in polynomial space [9]. The interesting case is for formulas of the form $E\psi$. The proof of Lemma 17 provides a model-checking procedure for such formulas, but building the full automaton $\mathcal{A}_\psi$ and powerset model $\widehat{M}$ takes exponential space. We tackle this by building them both on the fly. The marking procedure of maximal history subformulas in states of $\widehat{M}$ is replaced with recursive calls to the model-checking procedure for CTLK^*_1, and by Lemma 17 we can implement in polynomial space a counter that indicates when the nondeterministic search of a satisfying path can be stopped.

▶ **Proposition 18.** *Model checking* CTLK^*_k *formulas of alternation depth one is in* PSPACE.

Proof. We define algorithm $\mathrm{MC}^1(\varphi, M, s, \langle I_a\rangle_{a\in Ag})$ that takes as input a CTLK^* history formula φ of alternation depth at most one, a model M, a state s and an information set I_a for each agent a, and returns true if $M, \tau \models \varphi$ for all histories τ such that $\mathrm{lst}(\tau) = s$ and $I_a(\tau) = I_a$ for all $a \in Ag$, and false otherwise. The algorithm is defined by induction on φ:

$\boldsymbol{\varphi = p}$: `return` $p \in V(s)$
$\boldsymbol{\varphi = \varphi_1 \vee \varphi_2}$: `return` $\mathrm{MC}^1(\varphi_1, M, s, \langle I_a\rangle_{a\in Ag})$ or $\mathrm{MC}^1(\varphi_2, M, s, \langle I_a\rangle_{a\in Ag})$
$\boldsymbol{\varphi = \neg\varphi'}$: `return` not $\mathrm{MC}^1(\varphi, M, s, \langle I_a\rangle_{a\in Ag})$
$\boldsymbol{\varphi = K_b\varphi'}$: `return` $\mathrm{And}_{s'\in I_b}\mathrm{MC}^1(\varphi', M, s', \langle I'_a\rangle_{a\in Ag})$, where $I'_b = I_b$ and $I'_a = \emptyset$ for $a \neq b$.
$\boldsymbol{\varphi = E\psi}$: Let $\mathrm{MaxSub}(\psi)$ be the set of maximal history subformulas of ψ, let ψ' be the LTL formula obtained from ψ by considering subformulas in $\mathrm{MaxSub}(\psi)$ as atoms, and let $\mathrm{Cl}(\psi)$ be the closure of $\mathrm{Sub}(\psi')$ under negation. First, guess a subset $S_1 \subseteq \mathrm{Cl}(\psi)$ of formulas that currently hold, in state s with information sets $\langle I_a\rangle_{a\in Ag}$. Check boolean consistency, i.e., check that the following two conditions hold:

- $\varphi_1 \vee \varphi_2 \in S_1$ iff $\varphi_1 \in S_1$ or $\varphi_2 \in S_1$
- $\neg\varphi' \in S_1$ iff $\varphi' \notin S_1$

Check that $\psi \in S_1$. Also, check that the truth of maximal history subformulas was guessed correctly: for all $\varphi' \in \mathrm{MaxSub}(\psi) \cap S_1$, check that $\mathrm{MC}^1(\varphi', M, s, \langle I_a\rangle_{a\in Ag})$, and for all $\varphi' \in \mathrm{MaxSub}(\psi) \setminus S_1$, check that $\mathrm{MC}^1(\neg\varphi', M, s, \langle I_a\rangle_{a\in Ag})$.
Now by Lemma 17 we know that if there exists a path that satisfies ψ, there exists an ultimately periodic one with start index and period less than $|M|2^{|Ag||M|+|\psi|}$. So let us guess $n_1, n_2 \leq |M|2^{|Ag||M|+|\psi|}$, representing respectively the start index and the period of the ultimately periodic path that the algorithm is going to guess. Set a counter c to zero.
While $\mathsf{c} < n_1$, do:

```
Step {
    guess $s' \in R(s)$
    $s := s',\ I_a := \mathrm{Up}^{\mathrm{sy}}(I_a, s')$
    guess a set $S_2 \subseteq \mathrm{Cl}(\psi)$
    check boolean consistency of $S_2$
    check dynamic consistency of $S_1$ and $S_2$:
        $X\varphi' \in S_1$ iff $\varphi' \in S_2$, and
        $\varphi_1 U \varphi_2 \in S_1$ iff $\varphi_2 \in S_1$, or ($\varphi_1 \in S_1$ and $\varphi_1 U \varphi_2 \in S_2$)
    inductively check the truth of maximal history subformulas in $S_2$
    $S_1 := S_2, \mathsf{c} := \mathsf{c} + 1$
```

Once $\mathsf{c} = n_1$, let $S_{\mathrm{period}} := S_1$, $s_{\mathrm{period}} := s$, $I_{a_{\mathrm{period}}} := I_a$ for each $a \in Ag$, and $\mathsf{c} := 0$.
While $\mathsf{c} < n_2$, do:

- Mark which eventualities (formulas of the form $\varphi_1 U \varphi_2$) in S_{period} are satisfied
- Execute Step

Once $\mathsf{c} = n_2$, check that $s = s_{\text{period}}$ and $I_a = I_{a\text{period}}$ for all a. If it is the case we indeed guessed an ultimately periodic path in the powerset model, and we just need to check that all eventualities in S_{period} have been satisfied somewhere in the period. Return true if it is the case, false otherwise.

Since the algorithm simply follows the alternative semantics, its correctness follows from Proposition 16, together with Lemma 17 for the bound on the length of paths. ■

4.2 Reducing alternation

We now describe how we can use the powerset construction to eliminate one level of alternation of knowledge operators. This is done with a classic procedure that we recall for completeness.

▶ **Proposition 19.** *Given a* CTLK^* *formula* Φ *of alternation depth* $k+1$ *and a model* $\models$*, one can build a model* M' *of size at most* $|M|2^{|Ag||M|}$ *and a* CTLK^* *formula* Φ' *of alternation depth* k *and size* $|\Phi'| \leq |\Phi|$ *such that* $M \models \Phi$ *iff* $M' \models \Phi'$.

The construction of M' is as follows. First, build the powerset model $\widehat{M}$ as in Definition 15. In this model, history formulas of alternation depth one can be evaluated positionally, as stated in Proposition 16. Let $\text{Sub}_1(\Phi)$ be the set of maximal such formulas in $\text{Sub}(\Phi)$. For each formula φ in $\text{Sub}_1(\Phi)$ and each state $\widehat{s}$ of $\widehat{M}$, evaluate whether $\widehat{s} \models_I \varphi$ (since $\models_I$ is the memoryless semantics of CTLK^*, this can be done in PSPACE [18]), and mark state $\widehat{s}$ with the fresh atomic proposition p_φ if $\widehat{s} \models_I \varphi$. We abuse notation and still call $\widehat{M}$ the model obtained after this marking procedure (and similarly, $\widehat{s}$, $\widehat{\tau}$ and $\widehat{\pi}$ refer to states, histories and paths in the marked model). Also, for every subformula φ of Φ, define $\widehat{\varphi}$ by replacing each φ' in $\text{Sub}_1(\varphi)$ with atom $p_{\varphi'}$ (note that if φ contains no knowledge operator then $\widehat{\varphi} = \varphi$).

Unlike in Proposition 16, where we use the alternative memoryless semantics to evaluate positionally formulas of alternation depth one, this time we interpret $\widehat{\varphi}$ on $\widehat{M}$ with the perfect-recall semantics. One can prove the following lemma:

▶ **Lemma 20.** *For every history subformula* φ *and path subformula* ψ *of* Φ*, every history* τ *and path* π *in* M*,*

$$\begin{aligned} \tau \models \varphi \quad &\textit{iff} \quad \widehat{\tau} \models \widehat{\varphi} \\ \pi, n \models \psi \quad &\textit{iff} \quad \widehat{\pi}, n \models \widehat{\psi} \end{aligned}$$

Proof. The proof is by induction on formulas. Again we only treat the cases of atomic propositions, path quantifier and knowledge operators, all remaining cases follow directly by definition of the semantics and application of the induction hypothesis.

$\varphi = \boldsymbol{p}$: By Lemma 14, $\widehat{\tau}$ ends in state $(\text{lst}(h), \langle I_a(\tau)\rangle_{a\in Ag})$, so by definition of $\widehat{M}$ we have $\widehat{V}(\text{lst}(\widehat{\tau})) = V(\text{lst}(\tau))$, and we conclude by noting that $\widehat{p} = p$.

$\varphi = \boldsymbol{A\psi}$: We observe that the set $\{\pi \mid \tau \preccurlyeq \pi\}$ is in bijection with the set $\{\widehat{\pi} \mid \widehat{\tau} \preccurlyeq \widehat{\pi}\}$; the result then follows by induction hypothesis and the fact that $\widehat{A\psi} = A\widehat{\psi}$.

$\varphi = \boldsymbol{K_a\varphi'}$: Let us write $\text{lst}(\widehat{\tau}) = (s, \langle I_a\rangle_{a\in Ag})$. We consider two cases.

- If $\text{ad}(K_a\varphi') = 1$, then $\widehat{\varphi} = p_\varphi$. Thus $\widehat{\tau} \models p_\varphi$ iff $\text{lst}(\widehat{\tau})$ has been marked with p_φ, which by construction is done iff $\text{lst}(\widehat{\tau}) \models_I \varphi$, which by Proposition 16 is equivalent to $\tau \models \varphi$, and we are done.
- If $\text{ad}(K_a\varphi') > 1$, then $\widehat{\varphi} = K_a\widehat{\varphi'}$. In this case $\tau \models K_a\varphi'$ iff

$$\text{for all } \tau' \approx_a \tau,\ \tau' \models \varphi', \tag{2}$$

which by induction hypothesis is equivalent to

$$\text{for all } \tau' \approx_a \tau,\ \widehat{\tau}' \models \widehat{\varphi}'. \tag{3}$$

Because all histories start in the initial state, there is a bijection between $\{\tau' \mid \tau' \approx_a \tau\}$ and $\{\widehat{\tau}' \mid \widehat{\tau}' \approx_a \widehat{\tau}\}$. Thus (3) can be rewritten as

$$\text{for all } \widehat{\tau}' \approx_a \widehat{\tau},\ \widehat{\tau}' \models \widehat{\varphi}', \tag{4}$$

which is equivalent to $\widehat{\tau} \models K_a\widehat{\varphi}'$, and we are done.

■

Using Proposition 19 for the inductive case and Proposition 18 for the base case, we easily prove the following by induction on k.

▶ **Theorem 21.** *Model checking* CTLK* *formulas of alternation depth at most $k+1$ is in k-*EXPSPACE*, both for synchronous and asynchronous perfect recall.*

5 Lower bounds

In this section, we establish the following result which provides lower bounds for model checking against CTLK*, matching the upper bounds of Theorem 21.

▶ **Theorem 22.** *For $k \in \mathbb{N}$, model checking of CTLK^*_{k+1} for both the SPR and APR semantics is k-*EXPSPACE*-hard even if the formula is assumed to be a* fixed *$\mathsf{LTLK}_{k+1} \bigcap \mathsf{CTLK}_{k+1}$ formula and the number of agents is 2.*

Theorem 22 is proved by a polynomial-time reduction from a suitable domino-tiling problem [3]. An instance $\mathcal{I}$ of such a problem is a tuple $\mathcal{I} = (C, \Delta, n, d_{in}, d_{acc})$, where C is a finite set of colours, $\Delta \subseteq C^4$ is a set of tuples $(c_{down}, c_{left}, c_{up}, c_{right})$ of four colours, called *domino-types*, $n > 0$ is a natural number encoded in *unary*, and $d_{in}, d_{acc} \in \Delta$ are domino-types. Given $k \in \mathbb{N}$, a *k-grid of $\mathcal{I}$* is a mapping $f : [0,\ell] \times [0, \mathit{Tower}(n,k)-1] \to \Delta$ for some $\ell \in \mathbb{N}$. Intuitively, a k-grid is a finite grid, where each row consists of $\mathit{Tower}(n,k)$ cells, and each cell contains a domino type. A *k-tiling of $\mathcal{I}$* is a k-grid f satisfying the following additional constraints:

Initialisation and Acceptance: $f(0,0) = d_{in}$ and $f(\ell, j) = d_{acc}$ for some $j \in [0, \mathit{Tower}(n,k)-1]$;

Row adjacency-requirement: two adjacent cells in a row have the same color on the shared edge: for all $(i,j) \in [0,\ell] \times [0, \mathit{Tower}(n,k)-2]$, $[f(i,j)]_{right} = [f(i,j+1)]_{left}$

Column adjacency-requirement: two adjacent cells in a column have the same color on the shared edge: for all $(i,j) \in [0,\ell-1] \times [0, \mathit{Tower}(n,k)-1]$, $[f(i,j)]_{up} = [f(i+1,j)]_{down}$.

Given $k \in \mathbb{N}$, the problem of checking the existence of a k-tiling for $\mathcal{I}$ is k-EXPSPACE-complete [3]. Hence, Theorem 22 directly follows from the following proposition.

▶ **Proposition 23.** *Let $k \geq 0$. There is a* fixed *formula φ_k of $\mathsf{LTLK}_{k+1} \bigcap \mathsf{CTLK}_{k+1}$ such that one can build, in time polynomial in the size of the given instance $\mathcal{I}$, a Kripke structure $M_{\mathcal{I},k}$ with two agents so that $\mathcal{I}$ has a k-tiling* iff *$M_{\mathcal{I},k} \models_{\mathrm{sy}} \varphi_k$ (resp., $M_{\mathcal{I},k} \models_{\mathrm{as}} \varphi_k$).*

Proof of Proposition 23 for the synchronous setting We first provide a proof of Proposition 23 for the synchronous setting. At the end of this section, we explain the easy adaptation for the asynchronous case. Fix $k \geq 0$. In the following, we assume that $k \geq 1$ (the proof of Proposition 23 for the case $k = 0$ being simpler). First, we define a suitable encoding of the k-grids by infinite words over the set AP of atomic propositions given by $\mathrm{AP} := \mathit{Main} \cup \mathit{Tags}$, where

- $\mathit{Main} := \Delta \cup \{\$_1, \ldots, \$_k, \$, \mathit{acc}, \bot, 0, 1\}$
- $\mathit{Tags} := \{\#_1, \#_2, \mathit{col}, \mathit{good}\} \cup \bigcup_{h=1}^{h=k} \{(h, =), (h, \mathit{inc}), (\mathit{inc}, h)\}$

The propositions in *Main* are used to encode the k-grids, while the propositions in *Tags*, whose meaning will be explained later, are used to mark in a suitable way the codes of k-grids. Essentially, the *unmarked* code of a k-grid f is obtained by concatenating the codes of the rows of f starting from the first row and adding the suffix acc^ω if f satisfies the acceptance requirement, and the suffix $\bot^\omega$ otherwise. The code of a row is in turn obtained by concatenating the codes of the row's cells starting from the first cell.

In the encoding of a cell of a k-grid, we keep track of the content of the cell together with a suitable encoding of the cell number which is a natural number in $[0, \mathit{Tower}(n, k) - 1]$. Thus, for all $1 \leq h \leq k$, we define the notions of *h-block* and *well-formed h-block*. Essentially, for $1 \leq h < k$, well-formed h-blocks are finite words over $\{\$_1, \ldots, \$_h, 0, 1\}$ which encode integers in $[0, \mathit{Tower}(n, h) - 1]$, while well-formed k-blocks are finite words over $\Delta \cup \{\$_1, \ldots, \$_k, 0, 1\}$ which encode the cells of k-grids. In particular, for $h > 1$, a well-formed h-block encoding a natural number $m \in [0, \mathit{Tower}(n, h) - 1]$ is a sequence of $\mathit{Tower}(n, h - 1)$ $(h - 1)$-blocks, where the i^{th} $(h-1)$-block encodes both the value and (recursively) the position of the i^{th}-bit in the binary representation of m. Formally, the set of (well-formed) h-blocks is defined by induction on h as follows:

Base Step: $h = 1$. The notions of 1-block and well-formed 1-block coincide, and a 1-block is a finite word bl of length $n + 3$ having the form $\mathit{bl} = \$_1 \tau \mathit{bit}_1 \ldots \mathit{bit}_n \$_1$ such that $\mathit{bit}_1, \ldots, \mathit{bit}_n \in \{0, 1\}$ and $\tau \in \{0, 1\}$ if $1 < k$, and $\tau \in \Delta$ otherwise. For all $1 \leq \ell \leq n$, we say that bit_ℓ is the ℓ^{th} bit of bl. The *content* of bl is τ, and the *index* of bl is the natural number in $[0, \mathit{Tower}(n, 1) - 1]$ (recall that $\mathit{Tower}(n, 1) = 2^n$) whose binary code is $\mathit{bit}_1 \ldots \mathit{bit}_n$. The 1-block bl is *initial* (resp., *final*) if $\mathit{bit}_i = 0$ (resp., $\mathit{bit}_i = 1$) for all $1 \leq i \leq n$.

Induction Step: $1 < h \leq k$. An *h-block* is a finite word bl having the form $\$_h \tau \mathit{bl}_0 \ldots \mathit{bl}_j \$_h$ such that $j > 0$, $\mathit{bl}_0, \ldots, \mathit{bl}_j$ are $(h - 1)$-blocks, and $\tau \in \{0, 1\}$ if $h < k$, and $\tau \in \Delta$ otherwise. Additionally, we require that bl_0 is initial, bl_j is final, and for all $0 < i < j$, bl_i is not final. The *content* of bl is τ. The h-block bl is *initial* (resp., *final*) if the content of bl_i is 0 (resp., 1) for all $0 \leq i \leq j$. The h-block bl is *well-formed* if additionally, the following holds: $j = \mathit{Tower}(n, h - 1) - 1$ and for all $0 \leq i \leq j$, bl_i is well-formed and has index i. If bl is well-formed, then its *index* is the natural number in $[0, \mathit{Tower}(n, h) - 1]$ whose binary code is given by $\mathit{bit}_0, \ldots, \mathit{bit}_j$, where bit_i is the content of the sub-block bl_i for all $0 \leq i \leq j$.

Encoding of k-grids A *row-code* is a finite word $w_r = \$\mathit{bl}_0 \ldots \mathit{bl}_j\$$ satisfying the following:

- $\mathit{bl}_0, \ldots, \mathit{bl}_j$ are k-blocks;
- bl_0 is initial and bl_j is the unique final k-block.

The row-code w_r is *well-formed* if additionally, $j = \mathit{Tower}(n, k) - 1$ and for all $0 \leq i \leq j$, bl_i is well-formed and has index i. A *k-grid code* (resp., *well-formed k-grid code*) is an infinite word over AP of the form $w \cdot \tau^\omega$ such that (i) w is a finite sequence of row-codes (resp., well-formed row-codes), and (ii) $\tau = \mathit{acc}$ if the last row-code of w contains a block whose content is d_{acc} (acceptance), and $\tau = \bot$ otherwise. A k-grid code is *initialised* if the first k-block of the first row-code is d_{in}. Note that while k-grid codes encode grids of $\mathcal{I}$ having rows of arbitrary length, *well-formed* k-grid codes encodes the k-grids of $\mathcal{I}$. In particular, there is exactly one well-formed k-grid code associated with a given k-grid of $\mathcal{I}$.

Construction of $M_{\mathcal{I},k}$ in Proposition 23 for the synchronous case For the fixed $k \geq 1$, we now illustrate the construction of the finite Kripke structure $M_{\mathcal{I},k}$ over two agents, say a_1 and a_2, in Proposition 23. Essentially, $M_{\mathcal{I},k}$ nondeterministically generates all the *initialised* k-grid codes with the additional ability of nondeterministically marking some positions with the propositions in *Tags*. The main idea is to decompose the verification that a k-grid code is well-formed and encodes a k-tiling in layers implementable with polynomially many states of $M_{\mathcal{I},k}$, and invoking other layers thanks to the knowledge modalities for the two agents a_1 and a_2. In particular, the propositions in *Main* are observable by both agents, while the tag propositions in $\mathit{Tags} \setminus \{\#_1, \#_2\}$ are *not* observable by any agent. For the remaining two tag propositions $\#_1$ and $\#_2$, $\#_1$ (resp., $\#_2$) is observable by agent a_1 (resp., a_2) but not observable by agent a_2 (resp., a_1).

We now define the marking performed by the Kripke structure $M_{\mathcal{I},k}$. For a word w over 2^{AP}, the *content* of w is the word over $2^{\mathrm{AP}\setminus \mathit{Tags}}$ obtained by removing from each letter in w the propositions in *Tags*. Let $1 \leq h \leq k$. A *tagged h-block* is a word bl over 2^{AP} whose content is an h-block and:

- the initial position of bl is marked by the tag $\#_1$ if h is odd, and the tag $\#_2$ otherwise;
- if $h = 1$, then there is $1 \leq \ell \leq n$ such that the ℓ^{th} bit of bl is marked by the tag $\#_2$;
- if $h > 1$, there is exactly one $(h-1)$-sub-block sb of bl whose first position is marked by the tag $\#_2$ if h is odd, and the tag $\#_1$ otherwise;
- no other position of bl is marked.

A *simple tagged h-block bl* is defined in a similar way but we require that only the first position of bl is marked.

The initialised k-grid codes are marked by the Kripke structure $M_{\mathcal{I},k}$ as follows. A *tagged k-grid code* is an infinite word ν over 2^{AP} such that the content of ν is an *initialised* k-grid code and one of the following holds.

$(h,=)$-tagging with $1 \leq h \leq k$: there are two *tagged* h-blocks bl and bl' along ν such that bl' follows bl and:

- if $h = 1$, then the marked bit of bl has the same position as the marked bit of bl';
- each position in ν following the last position of bl' is marked by the tags in O where $\{(h,=)\} \subseteq O \subseteq \{(h,=), \mathit{good}\}$, and $\mathit{good} \in O$ iff *either* $h = 1$ and the marked bit of bl has the same value as the marked bit of bl', *or* $h > 1$ and the marked sub-block of bl and the marked sub-block of bl' have the same content.
- no other position of ν is marked.

$(h, \boldsymbol{\mathit{inc}})$-tagging with $1 \leq h \leq k$: there are two *tagged* h-blocks bl and bl' along ν such that bl' follows bl and:

- if $h = 1$, then the marked bit of bl has the same position as the marked bit of bl';
- bl and bl' are adjacent within the same $(h+1)$-block if $h < k$, and within the same row-code otherwise;
- each position in ν following the last position of bl' is marked by the tags in O where $\{(h, \mathit{inc})\} \subseteq O \subseteq \{(h, \mathit{inc}), \mathit{good}\}$, and:
  - case $h = 1$: let i_0 be the position of the least significant (i.e., rightmost) bit of bl whose value is 0 (note that since bl and bl' are adjacent, our encoding ensures that such a bit exists). Then, we require that $\mathit{good} \in O$ if and only if the marked bits of bl and bl' have the same value if the marked bit of bl precedes the i_0^{th} bit of bl, and the marked bits of bl and bl' have distinct value otherwise.
  - case $h > 1$: let sb_0 be the last $(h-1)$-sub-block of bl whose content is 0 (since bl and bl' are adjacent, our encoding ensures that such a sub-block sb_0 exists). Then, we require that $\mathit{good} \in O$ if and only if the marked sub-blocks of bl and bl' have the same content if the marked sub-block of bl precedes sb_0, and the marked sub-blocks of bl and bl' have distinct content otherwise.

- no other position of ν is marked.

simple (inc, h)-tagging with $1 \leq h \leq k$: there are two *simple tagged* h-blocks bl and bl' along ν such that bl' follows bl and:

- bl and bl' are adjacent within the same $(h+1)$-block if $h < k$, and within the same row-code otherwise;
- case $h < k$: each position in ν following the last position of bl' is marked by the tag (inc, h);
- case $h = k$: each position in ν following the last position of bl' is marked by the tags in O where $\{(inc, k)\} \subseteq O \subseteq \{(inc, k), good\}$. Moreover, $good \in O$ iff $[d]_{right} = [d']_{left}$, where $d \in \Delta$ (resp., $d' \in \Delta$) is the content of bl (resp., bl');
- no other position of ν is marked.

column-tagging: there are two *simple tagged* k-blocks bl and bl' along ν such that bl' follows bl and:

- bl and bl' belong to two adjacent row-codes in ν;
- each position in ν following the last position of bl' is marked by the tags in O where $\{col\} \subseteq O \subseteq \{col, good\}$, and $good \in O$ iff $[d]_{up} = [d']_{down}$, where $d \in \Delta$ (resp., $d' \in \Delta$) is the content of bl (resp., bl');
- no other position of ν is marked.

A *partial tagged k-grid code* is the prefix of some tagged k-grid code whose last position is labelled by tags in $Tags \setminus \{\#_1, \#_2\}$. Thus, we have four different types of partial tagged k-grid codes ρ, where a type is identifiable by the tag proposition in $Tags \setminus \{\#_1, \#_2, good\}$ which marks the last position of ρ. The additional proposition $good$ is used to check whether some additional condition is fulfilled depending on the specific type. Intuitively, partial $(h, =)$-tagged k-grid codes are exploited as nested layers for checking that distinct well-formed h-blocks along the given initialised k-grid code have the same index, while partial (h, inc)-tagged k-grid codes are used as nested layers to check that the indices of adjacent h-blocks bl_1 and bl_2 are *consecutive* (i.e, bl_1 is not final and the index of bl_2 is the index of bl_1 plus one). Finally, partial simple (inc, h)-tagged k-grid codes and partial column-tagged k-grid-codes are exploited as first-level layers for verifying well-formedness and the row adjacency-requirement and column adjacency-requirement.

Let M be a Kripke structure over AP with valuation function V. Given a history τ (resp., a path π), the *trace* of τ (resp., π) is the finite (resp., infinite word) over 2^{AP} given by $V(\tau_0)\ldots V(\tau_{m-1})$ where $m = |\tau|$ (resp., $V(\pi_0)V(\pi_1)\ldots$). A trace of M is a trace of some history or path in M. By construction, the following result is straightforward.

▶ **Lemma 24.** *Let $k \geq 1$. One can construct in time polynomial in the size of $\mathcal{I}$, a finite Kripke structure $M_{\mathcal{I},k} = (\mathrm{AP}, S, R, V, \{\sim_a\}_{a \in \{a_1, a_2\}}, s^\iota)$ over two agents a_1 and a_2 such that:*

1. *the set of finite traces of $M_{\mathcal{I},k}$ coincides with the set of prefixes of* tagged *k-grid codes;*[2]
2. *the set of infinite traces of $M_{\mathcal{I},k}$ contains the set of initialised k-grid codes;*
3. *for all $i = 1, 2$ and states s and s', $s \sim_{a_i} s'$* iff $V(s) \cap (Main \cup \{\#_i\}) = V(s') \cap (Main \cup \{\#_i\})$.

Construction of the fixed formula φ_k in Proposition 23 A *K-propositional formula* is a CTLK^* formula which only contains the knowledge modalities, Boolean connectives, and atomic propositions. For each $h \in \mathbb{N}$, a *K_h-propositional formula* is a K-propositional formula with alternation depth h. Let $k \geq 1$ and $M_{\mathcal{I},k}$ be the Kripke structure of Lemma 24. By

[2] note that the prefix of an initialised k-grid code is always the prefix of some *tagged* k-grid code

Lemma 24, the indistinguishability relations $\sim_{a_i}$ of $M_{\mathcal{I},k}$ (for $i = 1, 2$) depend only on the valuation function. Hence, histories which have the same trace are indistinguishable by any agent and satisfy the same K-propositional formulas. Thus, for a finite trace ρ of $M_{\mathcal{I},k}$ and a K-propositional formula, we write $\rho \models \psi$ to mean that $\tau \models \psi$ under the SPR semantics for any history whose trace is ρ. The core result in the proposed reduction is represented by the following lemma which together with Lemma 24 concludes the proof of Proposition 23 for the SPR semantics.

▶ **Lemma 25.** *Let $k \geq 1$ and $M_{\mathcal{I},k}$ be the Kripke structure of Lemma 24. Then there are a* fixed $\mathsf{LTLK}_{k+1} \bigcap \mathsf{CTLK}_{k+1}$ *formula φ_k and for all $1 \leq h \leq k$, two* fixed *K_h-propositional formulas $\varphi^h_=$ and φ^h_{inc} such that the following holds:*
1. *Let ρ be a partial $(h, =)$-tagged k-grid code. If the two tagged h-blocks bl and bl' of ρ are well-formed, then $\rho \models \varphi^h_=$* iff *bl and bl' have the same index.*
2. *Let ρ be a partial (h, inc)-tagged k-grid code. If the two tagged h-blocks bl and bl' of ρ are well-formed, and bl precedes bl', then $\rho \models \varphi^h_{inc}$* iff *the indices of bl and bl' are consecutive.*
3. *$M_{\mathcal{I},k} \models \varphi_k$* iff *there is a path of $M_{\mathcal{I},k}$ whose trace is a well-formed k-code encoding a k-tiling of $\mathcal{I}$.*

Proof. We first prove Properties 1 and 2. This is done by induction on $1 \leq h \leq k$. We also show that $\varphi^h_=$ (resp., φ^h_{inc}) is a fixed K_h-propositional formula of the form $K_{a_i}\psi$, where $i = 1$ if h is odd, and $i = 2$ otherwise. Here, we focus on Property 2 (the proof of Property 1 is similar).

Let ρ be a partial (h, inc)-tagged k-grid code such that the tagged h-blocks bl and bl' of ρ are well-formed and bl' follows bl.

For the base case, assume that $h = 1$. By construction, the first position of bl (resp., bl') is marked by $\#_1$, and the marked bit of bl (resp., bl') is marked by $\#_2$. Moreover, the last position of ρ is marked by the tags in O, where $\{(1, inc)\} \subseteq O \subseteq \{(1, inc), good\}$. Let Π be the set of partial $(1, inc)$-tagged k-grid codes ρ' having the same content as ρ and the same marked $\#_1$-positions as ρ (i.e., ρ and ρ' mark the same two adjacent 1-blocks). By construction, the indices of bl and bl' are consecutive if and only if for each $\rho' \in \Pi$, the last position of ρ' is marked by $good$. By Lemma 24, Π coincides with the set of finite traces of $M_{\mathcal{I},k}$ which are SPR a_1-indistinguishable from ρ and whose last position is tagged by $(1, inc)$. Hence, the fixed K_1-propositional formula φ^1_{inc} capturing Property 2 for $h = 1$ is given by $K_{a_1}((1, inc) \rightarrow good)$.

Now assume that $h > 1$ and h is odd (the case where h is even being similar). By construction, the first position of bl (resp., bl') is marked by $\#_1$, and the marked $(h-1)$-sub-block of bl (resp., bl') is marked by $\#_2$. Moreover, the last position of ρ is marked by the tags in O, where $\{(h, inc)\} \subseteq O \subseteq \{(h, inc), good\}$. Let Π be the set of partial (h, inc)-tagged k-grid codes ρ' having the same content as ρ and the same marked $\#_1$-positions as ρ (i.e., ρ and ρ' mark the same two adjacent h-blocks). By construction, the indices of bl and bl' are consecutive if and only if for each $\rho' \in \Pi$ such that the two $\#_2$-marked $(h-1)$-sub-blocks of ρ' have the same index, it holds that the last position of ρ' is marked by $good$. By Lemma 24, Π coincides with the set of finite traces of $M_{\mathcal{I},k}$ which are SPR a_1-indistinguishable from ρ and whose last position is tagged by (h, inc). Moreover, for each $\rho' \in \Pi$, the set of partial $(h-1, =)$-tagged k-grid codes which have the same content as ρ' and the same marked $\#_2$-positions as ρ' (i.e., the initial positions of the two $\#_2$-marked $(h-1)$-sub-blocks of ρ') coincides with the set of finite traces of $M_{\mathcal{I},k}$ which are SPR a_2-indistinguishable from ρ' and whose last position is tagged by $(h-1, =)$. Thus, by the induction hypothesis on Property 1, the fixed K_h-propositional formula φ^h_{inc} capturing Property 2 when h is odd is given by $K_{a_1}\big(((h, inc) \wedge K_{a_2}((h-1, =) \wedge \varphi_{=}^{h-1})) \rightarrow good\big)$. Note that since $h-1$ is even, by the induction hypothesis $\varphi_{=}^{h-1}$ is of the form $K_{a_2}\psi$. Hence, φ_{inc}^{h} has alternation depth h.

Proof of Property 3. We now illustrate the construction of the fixed $\mathsf{LTLK}_{k+1} \bigcap \mathsf{CTLK}_{k+1}$ formula φ_k ensuring Property 3 of Lemma 25. For this, by exploiting the K-propositional formulas $\varphi_{=}^{h}$ and φ_{inc}^{h}, we first construct, for all $2 \leq h \leq k$, a K_h-propositional formula φ_{bl}^{h}, and two K_{k+1}-propositional formulas φ_{row} and φ_{col} satisfying the following for each initialised k-grid code ν:

- if all the $(h-1)$-blocks in ν are well-formed, then $\nu_{\leq i} \models \varphi_{bl}^{h}$ for all $i \geq 0$ *iff* all the h-blocks in ν are well-formed too;
- if all k-blocks in ν are well-formed, then $\nu_{\leq i} \models \varphi_{row}$ for all $i \geq 0$ *iff* the row-codes in ν are well-formed and for all adjacent k-blocks bl and bl' in a row-code of ν such that bl' follows bl, $[d]_{right} = [d']_{left}$, where $d \in \Delta$ (resp., $d' \in \Delta$) is the content of bl (resp., bl').
- if ν is well-formed, then $\nu_{\leq i} \models \varphi_{col}$ for all $i \geq 0$ *iff* for all the k-blocks bl and bl' such that bl and bl' belong to two adjacent row-codes, bl' follows bl, and bl and bl' have the same index, it holds that $[d]_{up} = [d']_{down}$, where $d \in \Delta$ (resp., $d' \in \Delta$) is the content of bl (resp., bl').

Intuitively, the formulas $\varphi_{bl}^{2}, \ldots, \varphi_{bl}^{k}$, φ_{row}, and φ_{col} require that an initialised k-grid code is well-formed and satisfies the column adjacency-requirement and the row adjacency-requirement. First, let us define the formula φ_{bl}^{h} where $2 \leq h \leq k$. Assume that h is even (the other case where h is odd being similar). Let ν be an initialised k-grid code whose $(h-1)$-blocks are well-formed. In order to ensure that the h-blocks of ν are well-formed as well, we need to require that adjacent $(h-1)$-blocks of ν belonging to the same h-block have consecutive indices. For this, we exploit the K_{h-1}-propositional formula φ_{inc}^{h-1} and the simple $(inc, h-1)$-tagging. Since $h-1$ is odd, in a partial simple $(inc, h-1)$-tagging k-grid code ρ, the first positions of the adjacent tagged $(h-1)$-blocks of ρ are marked by $\#_1$, and the same holds for partial $(h-1, inc)$-tagged k-grid codes. Thus, formula φ_{bl}^{h} is defined as follows.

$$\varphi_{bl}^{h} := K_{a_2}\big((inc, h-1) \rightarrow K_{a_1}((h-1, inc) \rightarrow \varphi_{inc}^{h})\big)$$

By Lemma 24, under the SPR semantics, the knowledge modality K_{a_2} together with the tag proposition $(inc, h-1)$ allow to select all and only the partial simple $(inc, h-1)$-tagging k-grid codes which have the same content as the current prefix of ν, while the nested knowledge modality K_{a_1} and the tag proposition $(h-1, inc)$ allow to select all and only the partial $(h-1, inc)$-tagged k-grid codes which have the same content as the current prefix of ν. Hence, by Property 2, correctness of the construction directly follows. The definition of the K_{k+1}-propositional formulas φ_{row} and φ_{col} follows a similar pattern. Here, we focus on the construction of φ_{col} which exploits the column-tagging and the formula $\varphi_{=}^{k}$. Assuming that k is odd, φ_{col} is defined as follows:

$$\varphi_{col} := K_{a_2}\big([col \wedge K_{a_1}((k, =) \wedge \varphi_{=}^{k})] \rightarrow good\big)$$

Given an initialised well-formed k-grid code ν, the formula above asserts that for all partial column-tagged k-grid codes ρ having the same content as the current prefix of ν, if the two simple tagged k-blocks of ρ have the same index, then $[d]_{up} = [d']_{down}$, where d (resp., d') is the content of the first (resp., second) simple tagged-block of ρ. Thus, since in partial columns-tagged k-grid codes, the two simple tagged k-blocks belong to two adjacent row-codes, correctness of the construction directly follows.

Finally, the fixed $\mathsf{LTLK}_{k+1} \bigcap \mathsf{CTLK}_{k+1}$ formula φ_k ensuring Property 3 of Lemma 25 is defined as follows:

$$\varphi_k := E\Big(\big(\bigwedge_{t \in Tags} \neg t \wedge \varphi_{row} \wedge \varphi_{col} \wedge \bigwedge_{h=2}^{k} \varphi_{bl}^{h}\big)\ U\ acc\Big)$$

which by Lemma 24 ensures the existence of a path of $M_{\mathcal{I},k}$ whose trace is an initialised well-formed k-grid code encoding a k-tiling. This concludes the proof of Lemma 25. ∎

Proof of Proposition 23 for the asynchronous setting For the asynchronous case, we slightly modify the construction of model $M_{\mathcal{I},k}$ in Lemma 24 by incorporating a bit represented by a fresh atomic proposition p_b that is flipped at every transition and is observed by all agents. In such a way the resulting model $M'_{\mathcal{I},k}$ generates the same traces as $M_{\mathcal{I},k}$ (modulo p_b), and for all histories τ and τ', $\tau \approx_a^{\mathrm{as}} \tau'$ iff $\tau \approx_a^{\mathrm{s}} \tau'$ (the asynchronous and synchronous semantics coincide). Formally, let $M_{\mathcal{I},k} = (\mathrm{AP}, S, R, V, \{\sim_a\}_{a \in Ag}, s^{\iota})$. We define $M'_{\mathcal{I},k} = (\mathrm{AP} \cup \{p_b\}, S \times \{0,1\}, R', V', \{\sim'_a\}_{a \in Ag}, (s^{\iota}, 0))$ where

- $(s,i)R'(s',j)$ if sRs' and $j = 1 - i$
- $V'(s,i) = \begin{cases} V(s) & \text{if } i = 0 \\ V(s) \cup \{p_b\} & \text{otherwise} \end{cases}$
- $(s,i) \sim'_a (s',j)$ if $s \sim_a s'$ and $i = j$

It is clear that $M'_{\mathcal{I},k}$ generates the same traces as $M'_{\mathcal{I},k}$, modulo the valuations of p_b. It is also clear that all agents observe every step, so that the asynchronous and synchronous semantics coincide. Also, since the bit $i \in \{0,1\}$ is reflected by p_b, it is also the case that histories that have the same trace are indistinguishable for both agents and satisfy the same K-propositional formulas, so that all the reasoning in the proof of Lemma 25 still holds. Finally, since the formulas built for Lemma 25 do not mention p_b, we obtain that $M_{\mathcal{I},k} \models_{\mathrm{sy}} \varphi_k$ iff $M'_{\mathcal{I},k} \models_{\mathrm{as}} \varphi_k$, which concludes the proof of Proposition 23 for the asynchronous case.

6 Conclusion

In this work we settle the exact complexity of model checking epistemic temporal logics with synchronous and asynchronous perfect recall, a 20-year-old problem, by showing that it is $(k-1)$-EXPSPACE-complete for formulas of alternation depth at most $k \geq 1$. This almost closes the picture for the 96 logics identified by Halpern and Vardi in their seminal work on epistemic temporal logics [14], with the exception of the "no learning" assumption which has, up to our knowledge, never been studied in conjunction with model checking. As mentioned in the introduction, most cases seem to boil down to already known cases for bounded memory. The only exception is for synchronous bounded memory for CTLK, which we plan to investigate in order to finally close the full picture.

References

1 Rajeev Alur, Pavol Černỳ, and Swarat Chaudhuri. Model checking on trees with path equivalences. In *TACAS*, pages 664–678. Springer, 2007.

2 Guillaume Aucher. Supervisory control theory in epistemic temporal logic. In *AAMAS*, pages 333–340, 2014. URL: `http://dl.acm.org/citation.cfm?id=2615787`.

3 P. Van Emde Boas. The Convenience of Tilings. In *Complexity, Logic, and Recursion Theory*, pages 331–363. Marcel Dekker Inc, 1997.

4 Laura Bozzelli, Bastien Maubert, and Sophie Pinchinat. Uniform strategies, rational relations and jumping automata. *Inf. Comput.*, 242:80–107, 2015. URL: `http://dx.doi.org/10.1016/j.ic.2015.03.012`, `doi:10.1016/j.ic.2015.03.012`.

5 Krishnendu Chatterjee, Laurent Doyen, Thomas A. Henzinger, and Jean-François Raskin. Algorithms for omega-regular games with imperfect information. In *CSL*, pages 287–302, 2006.

6 Francien Dechesne and Yanjing Wang. To know or not to know: epistemic approaches to security protocol verification. *Synthese*, 177(1):51–76, 2010.

7 Cătălin Dima. Revisiting satisfiability and model-checking for ctlk with synchrony and perfect recall. In *CLIMA*, pages 117–131, 2009. URL: `https://doi.org/10.1007/978-3-642-02734-5_8`, `doi:10.1007/978-3-642-02734-5_8`.

8 E. Allen Emerson and Joseph Y. Halpern. "Sometimes" and "not never" revisited: on branching versus linear time temporal logic. *J. ACM*, 33(1):151–178, 1986.

9 E Allen Emerson and Chin-Laung Lei. Modalities for model checking: Branching time logic strikes back. *Sci. Comput. Program.*, 8(3):275–306, 1987.

10 Kai Engelhardt, Peter Gammie, and Ron Van Der Meyden. Model checking knowledge and linear time: PSPACE cases. In *LFCS*, pages 195–211. Springer, 2007.

11 Ronald Fagin, Joseph Y Halpern, Yoram Moses, and Moshe Vardi. *Reasoning about knowledge.* MIT press, 2004.

12 Joseph Y. Halpern and Kevin R. O'Neill. Anonymity and information hiding in multiagent systems. *J. Comput. Secur.*, 13(3):483–512, 2005. URL: `http://content.iospress.com/articles/journal-of-computer-security/jcs237`.

13 Joseph Y. Halpern, Ron van der Meyden, and Moshe Y. Vardi. Complete axiomatizations for reasoning about knowledge and time. *SIAM J. Comput.*, 33(3):674–703, 2004. URL: `https://doi.org/10.1137/S0097539797320906`, `doi:10.1137/S0097539797320906`.

14 Joseph Y Halpern and Moshe Y Vardi. The complexity of reasoning about knowledge and time. In *STOC*, pages 304–315. ACM, 1986.

15 Joseph Y. Halpern and Moshe Y. Vardi. The complexity of reasoning about knowledge and time. 1. Lower bounds. *J Comput. Syst. Sci.*, 38(1):195–237, 1989. `doi:10.1145/12130.12161`.

16 Xiaowei Huang, Qingliang Chen, and Kaile Su. The complexity of model checking succinct multiagent systems. In *IJCAI*, pages 1076–1082, 2015. URL: `http://ijcai.org/Abstract/15/156`.

17 Xiaowei Huang and Ron van der Meyden. The complexity of epistemic model checking: Clock semantics and branching time. In *ECAI*, pages 549–554, 2010.

18 Jeremy Kong and Alessio Lomuscio. Symbolic model checking multi-agent systems against CTL*K specifications. In *AAMAS*, pages 114–122, 2017. URL: `http://dl.acm.org/citation.cfm?id=3091147`.

19 Richard E. Ladner and John H. Reif. The logic of distributed protocols. In *TARK*, pages 207–222, 1986.

20 Alessio Lomuscio and Franco Raimondi. The complexity of model checking concurrent programs against CTLK specifications. In *AAMAS*, pages 548–550, 2006. URL: `https://doi.org/10.1145/1160633.1160733`, `doi:10.1145/1160633.1160733`.

21 Rohit Parikh and Ramaswamy Ramanujam. Distributed processes and the logic of knowledge. In *Logic of Programs*, pages 256–268, 1985.

22 Amir Pnueli. The temporal logic of programs. In *FOCS*, pages 46–57. IEEE, 1977.

23 Bernd Puchala. Asynchronous omega-regular games with partial information. In *MFCS*, pages 592–603, 2010.

24 Franco Raimondi. *Model checking multi-agent systems.* PhD thesis, University of London, 2006.

25 John H. Reif. The complexity of two-player games of incomplete information. *J. Comput. Syst. Sci.*, 29(2):274–301, 1984. `doi:10.1016/0022-0000(84)90034-5`.

26 Ron van der Meyden. Common knowledge and update in finite environments. *Inf. Comput.*, 140(2):115–157, 1998. URL: `https://doi.org/10.1006/inco.1997.2679`, `doi:10.1006/inco.1997.2679`.

27 Ron van der Meyden and Nikolay V. Shilov. Model checking knowledge and time in systems with perfect recall (extended abstract). In *FSTTCS*, pages 432–445, 1999.

28 Ron van der Meyden and Kaile Su. Symbolic model checking the knowledge of the dining cryptographers. In *CSFW*, pages 280–291, 2004.

29 Ron van der Meyden and Moshe Y Vardi. Synthesis from knowledge-based specifications. In *CONCUR*, pages 34–49. Springer, 1998.

30 Moshe Y Vardi and Pierre Wolper. Reasoning about infinite computations. *Inf. Comp.*, 115(1):1–37, 1994.