The Complexity of Model Checking Knowledge and Time

Laura Bozzelli , **Bastien Maubert** and **Aniello Murano**
Università degli Studi di Napoli "Federico II", Italy
lr.bozzelli@gmail.com, bastien.maubert@gmail.com, murano@na.infn.it

Abstract

We establish the precise complexity of the model checking problem for the main logics of knowledge and time. While this problem was known to be non-elementary for agents with perfect recall, with a number of exponentials that increases with the alternation of knowledge operators, the precise complexity of the problem when the maximum alternation is fixed has been an open problem for twenty years. We close it by establishing improved upper bounds for CTL* with knowledge, and providing matching lower bounds that also apply for epistemic extensions of LTL and CTL.

1 Introduction

A central aspect of multi-agent systems is that agents only have partial knowledge about the system [Parikh and Ramanujam, 1985]. 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 ongoing multi-agent systems, epistemic logics have been combined with temporal logics such as LTL [Pnueli, 1977], CTL and CTL* [Emerson and Halpern, 1986]. 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 [Ladner and Reif, 1986; Fagin *et al.*, 2004], information flow and cryptographic protocols [van der Meyden and Su, 2004; Halpern and O'Neill, 2005] and knowledge-based programs [van der Meyden and Vardi, 1998].

The satisfiability problem for this family of logics has been thoroughly studied in [Halpern and Vardi, 1986; Halpern and Vardi, 1989], 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 [Halpern *et al.*, 2004].

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 [Dechesne and Wang, 2010]).

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* [Kong and Lomuscio, 2017] and PTIME-complete for extensions of CTL [Raimondi, 2006]. For the synchronous setting, the situation is similar, but for extensions of CTL, the problem becomes PSPACE-complete [Engelhardt *et al.*, 2007; Huang and van der Meyden, 2010].

For agents with perfect recall, the problem is undecidable when common knowledge is part of the language [van der Meyden and Shilov, 1999]. 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 [van der Meyden and Shilov, 1999; Alur *et al.*, 2007; Dima, 2009; Aucher, 2014; Bozzelli *et al.*, 2015]. 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 [van der Meyden and Shilov, 1999]. For both the synchronous and asynchronous semantics, it is known to be in $(k-1)$-EXPSPACE for CTLK [Alur *et al.*, 2007], and in k-EXPTIME for CTLK* [Aucher, 2014; Bozzelli *et al.*, 2015]. 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

	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.

our PSPACE-completeness result for the fragment of LTLK with alternation depth one generalises that for synchronous LTLK with one agent proved in [Engelhardt *et al.*, 2007].

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 [Lomuscio and Raimondi, 2006; Huang *et al.*, 2015].

2 Preliminaries

Let $\mathbb{N}$ be the set of natural numbers. For all $n, k \in \mathbb{N}$, define $\mathit{Tower}(n, 0) = n$ and $\mathit{Tower}(n, k+1) = 2^{\mathit{Tower}(n,k)}$.

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. The sets of *history* formulas φ and *path* formulas ψ are defined as follows:

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

where $p \in \mathcal{AP}$ and $a \in Ag$, 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 the usual syntactic fragments LTLK and CTLK of CTLK*. LTLK consists of formulas of the form $A\psi$ or $\neg A\psi$ where path quantifiers in ψ are immediately preceded by knowledge modalities. 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 restrictions of LTLK, CTLK and CTLK*, respectively, 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 1 (Models)**.** *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 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 $I \subseteq S$, 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 $\sim_a$. For a history τ, $\mathrm{lst}(\tau)$ is the last state of τ.

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 2. *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 3. *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 $\mathit{Obs}_a(\tau) = \mathit{Obs}_a(\tau')$.*

Definition 4 (Semantics)**.** *Fix a model M. The SPR (resp., APR) semantics of a history formula φ on a history τ and a path formula ψ on a path π and a point in time $n \in \mathbb{N}$ is defined by induction as follows (Boolean cases are omitted):*

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

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. 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 formula φ, whether $M \models_{\mathrm{sy}} \varphi$ (resp., $M \models_{\mathrm{as}} \varphi$). In this work we prove the following result.

Theorem 1. *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.*

In the rest of this work, every time a definition or result holds for both semantics, we write $\approx_a$ instead of $\approx_a^{\mathrm{s}}$ or $\approx_a^{\mathrm{as}}$.

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 [Reif, 1984] to eliminate imperfect information from two-player games, and in [van der Meyden, 1998; van der Meyden and Shilov, 1999; Alur *et al.*, 2007] to model check variants of LTLK and CTLK. We show that it also can be used to model check CTLK^*.

3.1 Information Sets and Updates

An information set captures the set of states that an agent considers possible at a given moment. The following definition is common to synchronous and asynchronous perfect recall.

Definition 5. *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.

We now define two different update functions, one for the synchronous and one for the asynchronous case. 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.

Definition 6. *The* synchronous update *of an information set I_a for agent a with a new state s is*

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

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. 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 I, we let $\mathrm{Reach}_i^a(I) \supseteq I$ be the set of states reachable from I via steps invisible for a. We can now define the update as follows:

Definition 7. *The* asynchronous update *of an information set I_a for agent a with a new state s is*

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

This definition is an adaptation to our setting of the one in [Puchala, 2010], which considers two-player games. The following result follows directly by applying the definitions:

Lemma 2. *For every history $\tau \cdot s$, $I_a^{\mathrm{sy}}(\tau \cdot s) = \mathrm{Up}^{\mathrm{sy}}(I_a^{\mathrm{sy}}(\tau), s)$ and $I_a^{\mathrm{as}}(\tau \cdot s) = \mathrm{Up}^{\mathrm{as}}(I_a^{\mathrm{as}}(\tau), s)$.*

3.2 Powerset Construction

Given a model M we define a powerset model $\widehat{M}$ of exponential size in which formulas of alternation depth 1 can be evaluated positionally. States of $\widehat{M}$ contain, in addition to the current state in M, the current information set of each agent. This construction can be instantiated either for the synchronous or asynchronous semantics by choosing the appropriate update function for information sets. In the following we will often omit to specify which semantics is considered, because the construction and reasoning work for both.

Definition 8. *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 $b \in Ag$, $(s, \langle I_a \rangle_{a \in Ag}) \widehat{\sim}_b (s', \langle I'_a \rangle_{a \in Ag})$ if $s' \in I_b$ and $I_b = I'_b$*
- $\widehat{s}^\iota = (s^\iota, \langle \{s^\iota\} \rangle_{a \in Ag})$

Note that $|\widehat{M}| = |M| 2^{|Ag||M|}$. 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 in τ. The next lemma follows by definition of $\widehat{M}$, $\widehat{\tau}$ and application of Lemma 2.

Lemma 3. *For every history τ,*

$$\mathrm{lst}(\widehat{\tau}) = (\mathrm{lst}(\tau), \langle I_a(\tau) \rangle_{a \in Ag})$$

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

Definition 9. *Given a powerset model $\widehat{M}$, a state $\widehat{s} \in \widehat{S}$ and a path $\widehat{\pi}$, the alternative semantics of a CTLK^*_1 formula is defined inductively as follows (we omit Boolean cases):*

$$
\begin{array}{lll}
\widehat{s} \models_I p & \text{if} & p \in \widehat{V}(\widehat{s}) \\
\widehat{s} \models_I E\psi & \text{if} & \text{for some } \widehat{\pi} \in \mathrm{Path}(\widehat{s}),\ \widehat{\pi} \models_I \psi \\
\widehat{s} \models_I K_a\varphi & \text{if} & \text{for all } \widehat{s}' \text{ s.t. } \widehat{s}' \widehat{\sim}_a \widehat{s},\ \widehat{s}' \models_I \varphi
\end{array}
$$

$\widehat{\pi} \models_I \varphi$ if $\widehat{\pi}_0 \models_I \varphi$
$\widehat{\pi} \models_I X\psi$ if $\widehat{\pi}_{\geq 1} \models_I \psi$
$\widehat{\pi} \models_I \psi_1 U \psi_2$ if $\exists i \geq 0$ such that $\widehat{\pi}_{\geq i} \models_I \psi_2$ and $\forall j$ such that $0 \leq j < i$, $\widehat{\pi}_{\geq j} \models_I \psi_1$

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 $\tau \approx_a \tau'$, then agent a knows the same things in τ and in τ'.

Proposition 4. *For every history formula $\varphi \in \mathsf{CTLK}_1^*$, each model M and history τ, $\tau \models \varphi$ iff $\mathrm{lst}(\widehat{\tau}) \models_I \varphi$.*

4 Upper Bounds

In this section we establish the upper bounds in Theorem 1. All the reasoning in this section is independent of the chosen semantics, synchronous or asynchronous perfect recall. We first show how to solve the case of alternation depth one in PSPACE, and then how to eliminate one level of alternation at the cost of an exponential blowup in the size of the model.

4.1 Alternation Depth One

Our PSPACE model-checking procedure contains three main ingredients. The first one is the on-the-fly construction and resolution of Büchi automata for LTL formulas [Vardi and Wolper, 1994]. We combine it with an on-the-fly construction of the powerset model, already used in [Alur *et al.*, 2007] for a variant of CTLK, which allows us to evaluate LTLK formulas of alternation depth one positionally. Finally, we use the meta-algorithm by Emerson and Lei [Emerson and Lei, 1987] to extend this procedure from LTLK to CTLK*.

In the main model-checking procedure, formulas of the form $E\psi$ are dealt with by guessing a path in the powerset model and evaluating ψ on it. 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 this can be done.

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 5. *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 sketch. We turn ψ into an LTL formula ψ' by evaluating and marking its maximal history subformulas in the powerset model $\widehat{M}$. We then build a nondeterministic Büchi word automaton $\mathcal{A}_{\psi'}$ that accepts exactly the models of ψ' and has at most $2^{|\psi'|}$ states [Vardi and Wolper, 1994]. 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, 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. $|M|2^{|Ag||M|+|\psi|}$. □

Proposition 6. *Model checking CTLK_1^* is in* PSPACE.

Proof sketch. We 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 [Emerson and Lei, 1987]. The interesting case is for formulas of the form $E\psi$. The proof of Lemma 5 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 5 we can implement in polynomial space a counter that indicates when the nondeterministic search of a satisfying path can be stopped. □

4.2 Reducing Alternation

We show how to use the powerset construction to eliminate one level of alternation of knowledge operators.

Proposition 7. *Given a CTLK* formula Φ of alternation depth $k+1$ and a model M, 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*

$$M \models \Phi \text{ iff } M' \models \Phi'.$$

The construction of M' is as follows. First, build the powerset model $\widehat{M}$ as in Definition 9. In this model, history formulas of alternation depth one can be evaluated positionally, as stated in Proposition 4. Let $\mathrm{Sub}_1(\Phi)$ be the set of such formulas in $\mathrm{Sub}(\Phi)$. For each formula φ in $\mathrm{Sub}_1(\Phi)$ 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 [Kong and Lomuscio, 2017]), 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 $\mathrm{Sub}_1(\varphi)$ with atom $p_{\varphi'}$.

Unlike in Proposition 4, 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 8. *For every history subformula φ and every history τ in M, it holds that $\tau \models \varphi$ iff $\widehat{\tau} \models \widehat{\varphi}$.*

Using Proposition 7 for the inductive case and Proposition 6 for the base case, we prove by induction on k that:

Theorem 9. *For $k \in \mathbb{N}$, model checking of CTLK_{k+1}^* for both the SPR and APR semantics is in k-*EXPSPACE.

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 9.

Theorem 10. *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 10 is proved by a polynomial-time reduction from a suitable domino-tiling problem [Boas, 1997]. 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:

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: 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: 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 [Boas, 1997]. Hence, Theorem 10 directly follows from the following proposition.

Proposition 11. *Let $k \geq 0$. There is a* fixed *formula φ_k of $\mathsf{LTLK}_{k+1} \bigcap \mathsf{CTLK}_{k+1}$ s.t. one can build, in time polynomial in the size of $\mathcal{I}$, a Kripke structure $M_{\mathcal{I},k}$ with two agents so that $\mathcal{I}$ has a k-tiling* iff *$M_{\mathcal{I},k} \models_{\text{sy}} \varphi_k$ (resp., $M_{\mathcal{I},k} \models_{\text{as}} \varphi_k$).*

Proof of Proposition 11 for the synchronous setting We first provide a proof of Proposition 11 for the synchronous setting. At the end of this section, we explain the easy adaptation for the asynchronous case. Fix $k \geq 0$. We assume that $k \geq 1$ (the proof for the case $k = 0$ being simpler).

First, we define a suitable encoding of the k-grids by infinite words over the set of propositions $\text{AP} := \mathit{Main} \cup \mathit{Tags}$, where $\mathit{Main} := \Delta \cup \{\$_1, \ldots, \$_k, \$, \mathit{acc}, \bot, 0, 1\}$ and $\mathit{Tags} := \{\#_1, \#_2, \mathit{col}, \mathit{good}\} \cup \bigcup_{h=1}^{h=k} \{(h, =), (h, \mathit{inc}), (\mathit{inc}, h)\}$. 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.

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*, where for $h < k$, well-formed h-blocks are finite words over $\{\$_1, \ldots, \$_h, 0, 1\}$ encoding integers in $[0, \mathit{Tower}(n, h) - 1]$, while well-formed k-blocks are finite words over $\Delta \cup \{\$_1, \ldots, \$_k, 0, 1\}$ encoding 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.

The set of (well-formed) h-blocks is defined by induction on h as follows. A 0-block is a bit $b \in \{0, 1\}$. The *content* of b is b itself, and b is *initial* (resp., *final*) if $b = 0$ (resp., $b = 1$). For $1 \leq h \leq k$, an *h-block* is a finite word *bl* of the form $\$_h \tau sb_0 \ldots sb_\ell \$_h$ such that (i) $\ell > 0$, and $\ell = n - 1$ if $h = 1$, (ii) $\tau \in \{0, 1\}$ if $h < k$, and $\tau \in \Delta$ otherwise (τ is the *content* of *bl*), (iii) for all $1 \leq i \leq \ell$, sb_i is a $(h-1)$-block (the *i^{th} sub-block of bl*), and (iv) sb_1 is initial and sb_ℓ is the unique final sub-block. The block *bl* is *initial* (resp., *final*) if the content of each sub-block sb_i is 0 (resp., 1). The h-block *bl* is *well-formed* if *either* $h = 1$, *or* $h > 1$, $\ell = \mathit{Tower}(n, h - 1) - 1$ and for all $0 \leq i \leq \ell$, sb_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 $bit_0, \ldots, bit_\ell$, where bit_j is the content of sb_j for all $0 \leq j \leq \ell$.

Encoding k-grids A *row-code* $w_r = \$bl_0 \ldots bl_\ell \$$ is a finite word such that $bl_0, \ldots, bl_\ell$ are k-blocks, bl_0 is initial, and bl_ℓ is the unique final k-block. The row-code w_r is *well-formed* if additionally, $\ell = \mathit{Tower}(n, k) - 1$ and for all $0 \leq i \leq \ell$, 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 encode the k-grids of $\mathcal{I}$.

Construction of $M_{\mathcal{I},k}$ in Proposition 11 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 11. 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 $2^{\text{AP} \setminus \mathit{Tags}}$ 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;
- 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 model $M_{\mathcal{I},k}$ as follows. A *tagged k-grid code* is an infinite word ν over 2^{AP} s.t. the content of ν is an *initialised* k-grid code and there are two (simple) tagged h-blocks *bl* and bl' along ν for some $1 \leq h \leq k$ so that: (i) bl' follows *bl*, (ii) there is a non-empty

set $O \subseteq \mathit{Tags} \setminus \{\#_1, \#_2\}$ so that each position in ν following the last position of bl' is marked by the tags in O, (iii) no other position of ν is marked, and (iv) the following holds:

$(h,=)$**-tagging:** *bl* and bl' are *tagged* h-blocks, $\{(h,=)\} \subseteq O \subseteq \{(h,=), \mathit{good}\}$, and $\mathit{good} \in O$ iff the marked sub-block *sb* of *bl* and the marked sub-block sb' of bl' have the same content. Moreover, if $h = 1$, then *sb* (which is a marked bit) has the same position as sb'.

(h, inc)**-tagging:** *bl* and bl' are adjacent *tagged* h-blocks (i.e., adjacent within the same $(h+1)$-block if $h < k$, and within the same row-code otherwise), $\{(h, \mathit{inc})\} \subseteq O \subseteq \{(h, \mathit{inc}), \mathit{good}\}$, and $\mathit{good} \in O$ *iff* for the last $(h-1)$-sub-block sb_0 of *bl* whose content is 0, the marked sub-blocks of *bl* and bl' have the same content if the marked sub-block of *bl* precedes sb_0, and have distinct content otherwise. Moreover, if $h = 1$, then the marked sub-block of *bl* has the same position as the marked sub-block of bl'.

simple (inc, h)**-tagging**: *bl* and bl' are adjacent *simple tagged* h-blocks, $\{(\mathit{inc}, h)\} \subseteq O \subseteq \{(\mathit{inc}, h), \mathit{good}\}$. Moreover, if $h < k$, then $O = \{(\mathit{inc}, h)\}$. If instead $h = k$, then $\mathit{good} \in O$ iff $[d]_{\mathit{right}} = [d']_{\mathit{left}}$, where $d \in \Delta$ (resp., $d' \in \Delta$) is the content of *bl* (resp., bl').

column-tagging: $h = k$, *bl* and bl' are *simple tagged* k-blocks belonging to two adjacent row-codes in ν, $\{\mathit{col}\} \subseteq O \subseteq \{\mathit{col}, \mathit{good}\}$, and $\mathit{good} \in O$ iff $[d]_{\mathit{up}} = [d']_{\mathit{down}}$, where $d \in \Delta$ (resp., $d' \in \Delta$) is the content of *bl* (resp., bl').

A *partial tagged* k*-grid code* is the prefix of some tagged k-grid code whose last position is labelled by tags in $\mathit{Tags} \setminus \{\#_1, \#_2\}$. Thus, we have four different types of partial tagged k-grid codes ρ, where a type is identifiable by the tag proposition in $\mathit{Tags} \setminus \{\#_1, \#_2, \mathit{good}\}$ 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. A *trace* of M is a word ρ over 2^{AP} of the form $\rho = V(s_0)V(s_1)\ldots$ where $s_0 s_1 \ldots$ is a path or history of M (we say that ρ is the trace of $s_0 s_1 \ldots$). By construction, the following result is straightforward.

Lemma 12. *Let $k \geq 1$. One can construct in time polynomial in the size of $\mathcal{I}$, a finite Kripke structure $M_{\mathcal{I},k} = (\mathrm{AP}, S, R, V, \{\sim_a\}_{a \in \{a_1, a_2\}}, s^\iota)$ such that:*

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

Construction of the fixed formula φ_k in Proposition 11 A *K-propositional formula* (resp., *K_h-propositional formula* for $h \in \mathbb{N}$) is a CTLK^* (resp., CTLK_h) formula which does not have occurrences of temporal modalities and path quantifiers. By Lemma 12, histories of $M_{\mathcal{I},k}$ 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 ψ, we write $\rho \models \psi$ to mean that $\tau \models \psi$ under the SPR semantics for any history whose trace is ρ. We establish the following crucial technical lemma.

Lemma 13. *Let $k \geq 1$ and $M_{\mathcal{I},k}$ be the Kripke structure of Lemma 12. Then there are two* fixed *K_{k+1}-propositional formulas φ_{row} and φ_{col}, and for all $2 \leq h \leq k$, a* fixed *K_h-propositional formula φ^h_{bl}, satisfying the following for each initialised k-grid code ν:*

- *if the $(h-1)$-blocks in ν are well-formed, then $\nu_{\leq i} \models \varphi^h_{bl}$ for all $i \geq 0$* iff *the h-blocks in ν are well-formed;*
- *if all k-blocks in ν are well-formed, then $\nu_{\leq i} \models \varphi_{\mathit{row}}$ 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, $[d]_{\mathit{right}} = [d']_{\mathit{left}}$, where $d \in \Delta$ (resp., $d' \in \Delta$) is the content of bl (resp., bl').*
- *if ν is well-formed, then $\nu_{\leq i} \models \varphi_{\mathit{col}}$ 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 $[d]_{\mathit{up}} = [d']_{\mathit{down}}$, where $d \in \Delta$ (resp., $d' \in \Delta$) is the content of bl (resp., bl').*

Due to lack of space, the proof of Lemma 13 is omitted. Let φ_{row}, φ_{col}, φ^h_{bl} be the fixed K_{k+1}-propositional formulas of Lemma 13, and φ_k be the fixed $\mathsf{LTLK}_{k+1} \bigcap \mathsf{CTLK}_{k+1}$ formula $E\big((\bigwedge_{t \in \mathit{Tags}} \neg t \wedge \varphi_{\mathit{row}} \wedge \varphi_{\mathit{col}} \wedge \bigwedge_{h=2}^{k} \varphi^h_{bl})\ U\ \mathit{acc}\big)$. By Lemmata 12–13, $M_{\mathcal{I},k} \models_{\mathrm{sy}} \varphi_k$ *iff* there is a path of $M_{\mathcal{I},k}$ whose trace is a well-formed k-code encoding a k-tiling of $\mathcal{I}$, which concludes the proof of Proposition 11 for the synchronous case. For the asynchronous case, we slightly modify the construction of model $M_{\mathcal{I},k}$ in Lemma 12 by incorporating a bit represented by a fresh 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', $\tau \approx^{\mathrm{as}}_a \tau'$ iff $\tau \approx^{\mathrm{s}}_a \tau'$ (the asynchronous and synchronous semantics coincide). Lemma 13 holds for the new model as well. Hence, $M_{\mathcal{I},k} \models_{\mathrm{sy}} \varphi_k$ iff $M'_{\mathcal{I},k} \models_{\mathrm{as}} \varphi_k$ and we are done.

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 [Halpern and Vardi, 1986], with the exception of the "no learning" assumption which has, up to our knowledge, never been studied in conjunction with model checking. With no learning most cases seem to boil down to known cases of 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

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

[Aucher, 2014] Guillaume Aucher. Supervisory control theory in epistemic temporal logic. In *AAMAS*, pages 333–340, 2014.

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

[Bozzelli *et al.*, 2015] Laura Bozzelli, Bastien Maubert, and Sophie Pinchinat. Uniform strategies, rational relations and jumping automata. *Inf. Comput.*, 242:80–107, 2015.

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

[Dima, 2009] Cătălin Dima. Revisiting satisfiability and model-checking for ctlk with synchrony and perfect recall. In *CLIMA*, pages 117–131, 2009.

[Emerson and Halpern, 1986] 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.

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

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

[Fagin *et al.*, 2004] Ronald Fagin, Joseph Y Halpern, Yoram Moses, and Moshe Vardi. *Reasoning about knowledge*. MIT press, 2004.

[Halpern and O'Neill, 2005] Joseph Y. Halpern and Kevin R. O'Neill. Anonymity and information hiding in multiagent systems. *J. Comput. Secur.*, 13(3):483–512, 2005.

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

[Halpern and Vardi, 1989] 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.

[Halpern *et al.*, 2004] 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.

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

[Huang *et al.*, 2015] Xiaowei Huang, Qingliang Chen, and Kaile Su. The complexity of model checking succinct multiagent systems. In *IJCAI*, pages 1076–1082, 2015.

[Kong and Lomuscio, 2017] Jeremy Kong and Alessio Lomuscio. Symbolic model checking multi-agent systems against CTL*K specifications. In *AAMAS*, pages 114–122, 2017.

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

[Lomuscio and Raimondi, 2006] Alessio Lomuscio and Franco Raimondi. The complexity of model checking concurrent programs against CTLK specifications. In *AAMAS*, pages 548–550, 2006.

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

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

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

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

[Reif, 1984] John H. Reif. The complexity of two-player games of incomplete information. *J. Comput. Syst. Sci.*, 29(2):274–301, 1984.

[van der Meyden and Shilov, 1999] 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.

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

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

[van der Meyden, 1998] Ron van der Meyden. Common knowledge and update in finite environments. *Inf. Comput.*, 140(2):115–157, 1998.

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