Model Checking Temporal Epistemic Logic under Bounded Recall

Francesco Belardinelli,[1] **Alessio Lomuscio,**[1] **Emily Yu**[1,2]
[1]Imperial College London, UK
[2]Johannes Kepler University Linz, Austria

Abstract

We study the problem of verifying multi-agent systems under the assumption of bounded recall. We introduce the logic $\mathrm{CTLK}_{\mathrm{BR}}$, a bounded-recall variant of the temporal-epistemic logic CTLK. We define and study the model checking problem against CTLK specifications under incomplete information and bounded recall and present complexity upper bounds. We present an extension of the BDD-based model checker MCMAS implementing model checking under bounded recall semantics and discuss the experimental results obtained.

Introduction

An important aspect in the field of *multi-agent systems* (MAS) is the safety guarantee relating to the system correctness. Model checking (Clarke et al. 2018; Baier and Katoen 2008) has been put forward as a key technique for the formal verification of multi-agent systems (Gammie and van der Meyden 2004). A well-known difficulty with model checking is the *state explosion problem* (Clarke and Grumberg 1987), i.e., the fact that the size of the system state space grows exponentially in the number of state variables used to describe the system. Several approaches have focused on mitigating this difficulty. The leading methods include *abstractions* (Denning et al. 1989), *binary decision diagrams* (Bryant 1986), and *bounded model checking* (Biere et al. 1999). All of these have been employed in model checking MAS (Lomuscio, Qu, and Russo 2010; Penczek and Lomuscio 2003; Gammie and van der Meyden 2004).

A key difference between the use of model checking in hardware or reactive systems and in MAS is the specification languages typically employed. While in reactive and hardware systems, specifications of interest are generally limited to reachability or temporal properties, more expressive languages such as temporal-epistemic logics (Fagin et al. 1997) or logics for strategic abilities (Alur, Henzinger, and Kupferman 1997; Mogavero, Murano, and Vardi 2010) are normally considered for MAS.

In particular, a large body of research has focused on the verification of MAS against temporal-epistemic properties (van der Meyden and Shilov 1999; Lomuscio and Penczek 2007). In all these approaches agents in a MAS are assumed to have incomplete information, i.e., they do not perceive the whole of the global state.

Approaches, however, differ in terms of the underlying assumptions made in terms of the memory of the agents in the system. Tools such as MCMAS (Lomuscio, Qu, and Raimondi 2017) and Verics (Meski et al. 2014) focus on *observational semantics* only, i.e., an assumption is made that the successor local state of an agent depends on the present local state and the action taken by all the agents in the system. In contrast to this, MCK (Gammie and van der Meyden 2004) supports both observational semantics and a limited form of *recall*. Under the latter each agent in a run remembers all its past states up to that point. It follows that an agent's successor local state depends not only on her present local state, but on the whole of her local history up to that local state and the action performed by all agents in the system.

The perfect recall assumption is of interest in a number of applications including security, where the intruder is normally assumed to recall all messages that she has observed. Perfect recall, or memoryful semantics, is also widely used in game theoretical settings and beyond. The key difficulty in combing imperfect information and perfect recall is the resulting high complexity. It is known that model checking systems against temporal-epistemic specifications under incomplete information and perfect recall is NON-ELEMENTARY without the common knowledge modality (Dima 2008). The resulting complexity is so high that it inhibits any practical use of model checking.

To overcome this difficulty in this paper we put forward and study an approximation of perfect recall in terms of bounded recall. Specifically we consider agents as being endowed with bounded memory and their local state to be local histories of a finite and previously determined length. This work exploits the intuition that bounded recall semantics is equivalent to perfect recall whenever the properties to be expressed involve finite traces whose length is smaller than the size of the history.

The rest of the paper is organised as follows. In Section

2 we define a revised version of interpreted systems, we introduce the syntax and semantics CTLK$_{\mathrm{BR}}$, a variant of the temporal-epistemic logic CTLK under bounded recall, and define maps between interpreted systems with bounded and perfect recall. We continue in Section 3 by presenting results on the relationship between model checking perfect and bounded recall and show that model checking against CTLK$_{\mathrm{BR}}$ is PSPACE-COMPLETE. In Section 4 we introduce an implementation that we built supporting bounded recall and report the experimental results obtained when model checking agents under various sizes of recall.

Related Work. Model checking perfect recall against temporal-epistemic logics has extensively been studied from a theoretical standpoint (Dima 2008; Halpern and Vardi 1986; 1989; van der Meyden 1998). It has been shown that while the satisfiability problem for *Computation Tree Logic with knowledge* (CTLK) (Fagin et al. 1997) and *without common knowledge* under incomplete information and perfect recall is undecidable, the model checking problem for the same logic is decidable (Dima 2008). The authors also proved such model checking problem has an NON-ELEMENTARY upper bound. However, when common knowledge is added to the syntax, both the satisfiability and model checking problem under perfect recall are undecidable (van der Meyden 1998).

As stated above, most present approaches adopt a memoryless semantics to reduce the resulting complexity even if it limits the expressiveness considerably. The complexity of the model checking problem under memoryless semantics is well understood. Explicit model checking MAS against CTLK specifications is P-COMPLETE, whereas implicit model checking is PSPACE-COMPLETE (Lomuscio and Raimondi 2006). The contribution here presented differs from those above in that we consider a bounded semantics.

The work here presented is related to a recent direction of work investigating approximations of perfect recall to imperfect recall. For example, in (Jamroga, Knapik, and Kurpiewski 2017) and (Ågotnes and Walther 2009) an approximation of perfect recall in the context of ATL is put forward. In (Belardinelli, Lomuscio, and Malvone 2019; 2018), a three valued semantics is developed with the aim, again, of providing approximations for verifying ATL. In contrast to these works, we here work on an epistemic language. The method we propose for providing such approximation is entirely different from the ones cited above resulting in different properties.

We conclude by remarking that the implementation for bounded recall semantics we introduce is based on the existing model checker MCMAS (Lomuscio, Qu, and Raimondi 2017), which previously supported memoryless semantics only. The tool MCK (Gammie and van der Meyden 2004) also supports memoryless semantics, but, differently from MCMAS, it also supports perfect recall and clock semantics. However, for those cases under perfect recall it only supports systems with one agent only, which is a severe limitation in practice.

CTLK with Bounded Recall

In this section we present the syntax and semantics for the temporal-epistemic logic CTLK on systems under perfect and bounded recall, henceforth CTLK$_{\mathrm{PR}}$ and CTLK$_{\mathrm{BR}}$ respectively.

We first introduce interpreted system with k-bound memory. Interpreted Systems (Fagin et al. 1997) are a well-known formalism for reasoning about knowledge in multi-agent systems. Here we give a revised definition of such formalism, based on a notion of recall.

We use S^* to denote the set of finite (possibly empty) strings of elements of S.

Definition 1 (Interpreted System (with perfect recall))**.** *An interpreted system* $\mathcal{IS} = \langle (L_i, Act_i, P_i, t_i)_{i \in Ag}, I, \lambda \rangle$ *is defined on a set* $Agt = \{e, 1, \ldots, n\}$ *of agents such that for every* $i \in Agt$*,*

- L_i *is the set of* local observations.
- Act_i *is the set of* actions.
- $P_i : L_i^* \to (2^{Act_i} \setminus \emptyset)$ *is the* protocol function.
- $t_i : (L_i \times L_e)^* \times Act_e \times Act_1 \times \ldots \times Act_n \to 2^{L_i}$ *is the* local transition function *such that* $t_i(h_i, h_e, a_e, a_1, \ldots, a_n)$ *is defined only if* $a_i \in P(h_i)$ *and* $a_e \in P(h_e)$*.*
- $I \subseteq L_e \times L_1 \times \ldots \times L_n = S$ *is the set of global initial states.*
- $\lambda : AP \to 2^{S^*}$ *is the assignment function that assigns truth values to atomic propositions in histories.*

Differently from the standard notion of interpreted system (Fagin et al. 1997), in Definition 1 the protocol and transition functions are defined on histories of arbitrary length. Let $ACT = Act_e \times Act_1 \times \ldots \times Act_n$ be the set of joint actions. The *global transition function* $t : S^* \times ACT \to 2^S$ is defined such that $s \in t(h, a)$ iff for every $i \in Agt \cup \{e\}$, $s_i \in t_i(h_i, h_e, a)$.

The set $G \subseteq S^* = (L_e \times L_1 \times \ldots L_n)^*$ consists of the global histories reachable from the set I of initial global states by using the transition function t. Here we use $h[i]$ to denote the i-th component of a history h. With a slight abuse of notation we will often represent a global history $h \in S^*$ as a tuple $\langle h_e, h_1, \ldots, h_n \rangle$ of local histories such that for every $j \leq |h|$ and $i \in Agt \cup \{e\}$, $h[j]_i = h_i[j]$. Hereafter, given an infinite sequence $\rho \in S^\omega$ of states, we use $\rho[m, n]$ to denote the history from the m-th to the n-th position, for $m \leq n$. To describe the temporal evolution of an interpreted system under perfect and bounded recall, we introduce the notion of k-path. Here we use S^ω to denote the set of infinite strings of elements of S.

Definition 2 (k-path)**.** *Let* $k \in \mathbb{N} \cup \{\omega\}$*. A* k-path ρ *originating from history* h *is an infinite sequence* $s_1, s_2, \ldots$ *in* S^ω *such that*

- $\rho[1, |h|] = h$*;*
- *for all* $i \geq |h|$, $\rho_{i+1} = t(\rho[\max\{1, i - k\}, i], a)$ *for some joint action* a*, where* $(i - \omega) = 0$*.*

By Definition 2 in the case of no recall ($k = 0$) the transition function takes only the last visited state ρ_i as input to determine the successor state ρ_{i+1}. On the other hand, in the case of perfect recall ($k = \omega$) the transition function takes the whole history $\rho_1, \ldots, \rho_i$ so far to determine successor ρ_{i+1}.

To interpret epistemic operators, both individual and collective, for every $i \in Agt$, we introduce the *(k-bound) indistinguishability relation* $\sim_i^k$ on global histories such that $h \sim_i^k h'$ iff for all $j \leq k$, $h_i[|h| - min\{|h|, j\}] = h'_i[|h'| - min\{|h|, j\}]$, that is, the last k i-components of histories h and h' are equal.

Definition 3 (Indistinguishability relations)**.** *Let $A \subseteq Agt$ be a group of agents, and h, h' global histories.*

- *relation for everybody knows: $h \sim_A^{k,E} h'$ iff $\sim_A^{k,E}= (\bigcup_{a \in A} \sim_i^k)$, i.e., the local states of at least one agent in A are indistinguishable.*
- *relation for distributed knowledge: $h \sim_A^{k,D} h'$ iff $\sim_A^{k,D}= (\bigcap_{a \in A} \sim_i^k)$, i.e., the local states of all agents in A are indistinguishable.*

We now define the temporal-epistemic specification language that we will use in the rest of the paper. The syntax of CTLK_{PR} and CTLK_{BR} is the same as the standard setting for CTLK (Penczek and Lomuscio 2003).

Definition 4 (CTLK Syntax)**.** *A CTLK formula φ is constructed according to the following BNF:*

$$\begin{aligned} \varphi \quad ::= \quad & p \mid \neg\varphi \mid \varphi \wedge \varphi \mid EX\varphi \mid EG\varphi \mid E(\varphi U \varphi) \mid \\ & K_i\varphi \mid E_A\varphi \mid D_A\varphi \end{aligned}$$

The other Boolean connectives and CTL modalities can be introduced as usual. The readings of the epistemic modalities are as follows: $K_i\varphi$ denotes "agent i knows that φ holds"; $E_A\varphi$ denotes collective knowledge, *i.e.,* "each agent in group A knows that φ holds"; $D_A\varphi$ denotes distributed knowledge, *i.e.,* "all agents in group A together know that φ holds". Note that we do not employ a common knowledge modality.

The satisfaction of CTLK_{BR} and CTLK_{PR} is defined as follows.

Definition 5 (CTLK Satisfaction)**.** *Given an interpreted system $\mathcal{IS} = \langle (L_i, Act_i, P_i, t_i)_{i \in Agt}, I, \lambda \rangle$, history $h \in G$, and bound $k \in \mathbb{N} \cup \{\omega\}$:*

$$\begin{array}{ll}
\mathcal{IS}, h \models_k p & \textit{iff } h[\max\{1, |h| - k\}, |h|] \in \lambda(p) \\
\mathcal{IS}, h \models_k \neg\varphi & \textit{iff } \mathcal{IS}, h \not\models_k \varphi \\
\mathcal{IS}, h \models_k \varphi \wedge \psi & \textit{iff } \mathcal{IS}, h \models_k \varphi \textit{ and } \mathcal{IS}, h \models_k \psi \\
\mathcal{IS}, h \models_k EX\varphi & \textit{iff for some } k\textit{-path } \rho \textit{ originating from } h, \\
 & \quad \mathcal{IS}, \rho[max\{(|h| - k) + 1, 1\}, |h| + 1] \models_k \varphi \\
\mathcal{IS}, h \models_k EG\varphi & \textit{iff for some } k\textit{-path } \rho \textit{ originating from } h, \\
 & \quad \textit{for all } i \geq |h|, \\
 & \quad \mathcal{IS}, \rho[max\{i - k, 1\}, i] \models_k \varphi \\
\mathcal{IS}, h \models_k E(\varphi U \psi) & \textit{iff for some } k\textit{-path } \rho \textit{ originating from } h, \\
 & \quad \textit{for some } j \geq |h|, \\
 & \quad \mathcal{IS}, \rho[max\{j - k, 1\}, j] \models_k \psi \\
 & \quad \textit{and for all } |h| \leq i < j, \\
 & \quad \mathcal{IS}, \rho[max\{i - k, 1\}, i] \models_k \varphi \\
\mathcal{IS}, h \models_k K_i\varphi & \textit{iff } \mathcal{IS}, h' \models_k \varphi \textit{ for all } h' \sim_i^k h \\
\mathcal{IS}, h \models_k E_A\varphi & \textit{iff } \mathcal{IS}, h' \models_k \varphi \textit{ for all } h' \sim_A^{k,E} h \\
\mathcal{IS}, h \models_k D_A\varphi & \textit{iff } \mathcal{IS}, h' \models_k \varphi \textit{ for all } h' \sim_A^{k,D} h
\end{array}$$

We also write CTLK_{IR} to denote the logic CTLK interpreted on conventional observational semantics. In order to distinguish between CTLK_{BR} and CTLK_{PR}, we define the following:

Definition 6. *Given a CTLK formula φ, an interpreted system $\mathcal{IS}$, and a history $h \in G$, the satisfaction of* CTLK_{BR}*, $(\mathcal{IS}, h) \models_{BR} \varphi$ is defined as $(\mathcal{IS}, h) \models_k \varphi$ for a fixed natural $k \in \mathbb{N}$.*

The satisfaction of CTLK_{PR}*, $(\mathcal{IS}, h) \models_{PR} \varphi$ is defined as $(\mathcal{IS}, h) \models_\omega \varphi$.*

The satisfaction of CTLK_{IR}*, $(\mathcal{IS}, h) \models_{IR} \varphi$ is defined as $(\mathcal{IS}, h) \models_0 \varphi$.*

The following lemma states that, when considering the semantics on k-bounded recall, we may consider only histories of length at most k.

Lemma 1. *Under $k-$bounded recall, for every reachable history $h \in G$, we have $\mathcal{IS}, h \models_k \varphi$ iff $\mathcal{IS}, h[\max\{1, |h| - k\}, |h|] \models_k \varphi$.*

Proof. The proof is by induction on the structure of φ.

For atomic $\varphi = p \in AP$, $\mathcal{IS}, h \models_k \varphi$ iff $h[\max\{1, |h| - k\}, |h|] \in \lambda(p)$, iff $\mathcal{IS}, h[\max\{1, |h| - k\}, |h|] \models_k \varphi$.

The case for boolean connectives is immediate.

As regards $\varphi = EX\psi$, $\mathcal{IS}, h \models_k \varphi$ iff for some k-path ρ originating from h, $\mathcal{IS}, \rho[max\{(|h| - k) + 1, 1\}, |h| + 1] \models_k \varphi$. Notice that ρ is also a k-path originating from $h[\max\{1, |h| - k\}, |h|]$, and by induction hypothesis we obtain that $\mathcal{IS}, \rho[max\{(|h| - k) + 1, 1\}, |h| + 1] \models_k \varphi$. As a result, $\mathcal{IS}, h[\max\{1, |h| - k\}, |h|] \models_k \varphi$.

As regards $\varphi = K_i\psi$, $\mathcal{IS}, h \models_k \varphi$ iff $\mathcal{IS}, h' \models_k \psi$ for all $h' \sim_i^k h$. Notice that $h' \sim_i^k h$ iff $h'[\max\{1, |h'| - k\}, |h'|] \sim_i^k h[\max\{1, |h'| - k\}, |h'|]$, and by induction hypothesis $\mathcal{IS}, h'[\max\{1, |h'| - k\}, |h'|] \models_k \psi$. As a result, $\mathcal{IS}, h[\max\{1, |h| - k\}, |h|] \models_k \varphi$.

The cases for the other temporal and epistemic operators are similar. □

To illustrate the difference between bounded and perfect recall, consider the following variant of the train-gate-controller (TGC) scenario (van der Hoek and Wooldridge 2002). In this version the system consists of a controller and two trains, with each of the trains operating on a circular track. The two tracks share one tunnel, which has red-green traffic lights on both sides and allows only one train inside the tunnel at any time. The traffic lights are operated by the controller, and a train can enter the tunnel only when the light is green. The controller grants a train access randomly. However, if a train has requested to enter the tunnel, and has been waiting for three time stamps, then it will discard the request and exit the queue.

The scenario is easily expressible under perfect recall semantics as each train needs to recall the previous events before requesting to enter. Bounded recall can also be used as long as sufficient memory is employed. However, it is problematic to encode the scenario under memoryless semantics. In fact, consider the evaluation of the following CTLK formula:

$$\phi = \bigvee_{i=1}^{N} EF(waited_i\_3s \wedge EF(in\_tunnel_i))$$

The atomic proposition $waited_i\_3s$ means that train i has waited for three timestamps, and in_tunnel_i means train i is in the tunnel. This formula expresses the property that there exists a path in the system in which the train will have been waiting for three timestamps and the controller might grant access to the train later at some point, where N is the number of trains in the system and here $N = 2$. The formula can be verified under perfect recall, and if we use bounded recall semantics as an approximation, the verification result of the above property will depend on the window size that has been set. For example with a window size of 5, the train is able to recall the number of timestamps it has been waiting for.

The example illustrates the different semantics, the different capabilities and the fact that, intuitively, under a natural modelling, a specification may hold on one semantics and not on the other.

Model Checking CTLK$_{\mathbf{BR}}$

In this section we investigate the model checking problem for CTLK$_{\text{BR}}$ and CTLK$_{\text{PR}}$, and the relationship between bounded recall and perfect recall, by showing that if an existential formula φ holds under bounded recall then it also holds under perfect recall. Based on these properties, we can consider the bounded recall semantics as an approximation of perfect recall.

Definition 7 (Model Checking Problem)**.** *Given an interpreted system $\mathcal{IS}$, a CTLK formula φ, and a window size $k \in \mathbb{N} \cup \{\omega\}$, model checking $\mathcal{IS}$ against φ concerns deciding whether it is the case that $\mathcal{IS} \models_k \varphi$.*

The model checking problem for CTLK is normally investigated in the context of interpreted systems under imperfect recall (i.e., under observational semantics). In this case the complexity of the model checking problem is PTIME for models given explicitly (Lomuscio and Raimondi 2006) and PSPACE-COMPLETE for models given implicitly (Lomuscio and Raimondi 2006; Huang, Chen, and Su 2015), i.e., under compact representations such as the interpreted systems above.

Model checking interpreted systems with perfect recall is computationally more expensive due to the enriched expressiveness of these structures. Model checking against CTLK with common knowledge is undecidable as well as its satisfiability problem (van der Meyden 1998). Based on this, we can also get to the conclusion that explicit model checking with common knowledge is also undecidable.

Given an interpreted system $\mathcal{IS}$, under perfect recall semantics we consider histories of arbitrary length. For the same $\mathcal{IS}$ with a fixed memory bound $k \in \mathbb{N}$, by Lemma 1, given any history longer than k, we only need to consider its latest k observations, as the older ones are not relevant.

Definition 8 (ECTLK)**.** *The* positive existential *fragment of a CTLK formula φ is constructed as:*

$$\varphi \quad ::= \quad p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid EX\varphi \mid EG\varphi \mid E(\varphi U \varphi) \mid K_i\varphi \mid E_A\varphi \mid D_A\varphi$$

Hereafter we assume monotonicity on the interpretation of atoms, that is, if $h \in \lambda(p)$ then for all h' extending h, $h' \in \lambda(p)$.

Theorem 1. *Given an interpreted system $\mathcal{IS}$, an ECTLK formula φ, if $\mathcal{IS} \models_k \varphi$ then $\mathcal{IS} \models_{k+1} \varphi$.*

Proof. Here we use induction to prove this theorem.

Given $(\mathcal{IS}, h) \models_k p$, we have $h[\max\{1, |h| - k\}, |h|] \in \lambda(p)$ for some $k \in \mathbb{N}$. By monotonicity, $h \in \lambda(p)$ as well. Hence, $\mathcal{IS}, h \models_{k+1} p$. The case of Boolean connectives is immediate.

Given $(\mathcal{IS}, h) \models_k EX\varphi$, there exits an k-path ρ originating from h such that $\mathcal{IS}, \rho[max\{|h|+1-k, 1\}, |h|+1] \models_k \varphi$. By induction hypothesis, $\mathcal{IS}, \rho[max\{|h| - k, 1\}, |h| + 1] \models_{k+1} \varphi$. Thus $(\mathcal{IS}, h) \models_{k+1} EX\varphi$.

Given $(\mathcal{IS}, h) \models_k EG\varphi$, there exits an k-path ρ originating from h such that $\mathcal{IS}, \rho[max\{i - k, 1\}, i] \models_k \varphi$ for all $i \geq |h|$. Therefore, there exits a path ρ originating from h such that $\mathcal{IS}, \rho[max\{i-k-1, 1\}, i] \models_{k+1} \varphi$ for all $i \geq |h|$, and we get $(\mathcal{IS}, h) \models_{k+1} EG\varphi$.

We omit the proof for $E(\varphi U \phi)$, as it follows the same reasoning.

We assume $(\mathcal{IS}, h) \not\models_{k+1} K_i\varphi$, which means there exits a history h' such that $h \sim_i^{k+1} h'$ and $(\mathcal{IS}, h') \not\models_{k+1} \varphi$. In particular, for $k = |h|$, we have that $h \sim_i^k h'$ and $(\mathcal{IS}, h') \not\models_k \varphi$. This gives us $(\mathcal{IS}, h) \not\models_k K_i\varphi$.

The rest of the proof for E_A and D_A follows the same reasoning. □

Corollary 1. *Given an interpreted system $\mathcal{IS}$, an ECTLK formula φ, if $\mathcal{IS} \models_{BR} \varphi$ then $\mathcal{IS} \models_{PR} \varphi$.*

Proof. If we assume $\mathcal{IS}, h \models_k \varphi$ for some $k \geq 0$, we get $\mathcal{IS}, h \models_j \varphi$ for all $j \geq k$. When we set $j = \omega$, we get $\mathcal{IS}, h \models_{PR} \varphi$. □

Theorem 2. *If the bound $k = 0$, then $\mathcal{IS} \models_{BR} \varphi \Leftrightarrow \mathcal{IS} \models_{IR} \varphi$.*

Proof. This proof is straightforward. When $k = 0$, the state space under bounded recall semantics is exactly the same as that under imperfect recall, as all states have a length of 1. The verification of φ is the same under both semantics. □

We now explore the complexity results of model checking systems against CTLK$_{\text{BR}}$ and CTLK$_{\text{PR}}$. For the problem of determining whether $\mathcal{IS} \models_{BR} \varphi$, we claim that the complexity for a fixed bound k is in polynomial space (PSPACE) with respect to the size $|\mathcal{IS}|$ of the given interpreted system and the size $|\varphi|$ of the CTLK formula. The problem can be solved by recursive function calls with a depth of $|\varphi|$.

Theorem 3. *Model checking interpreted systems against CTLK specification under bounded recall is an EXPTIME problem. If the window size is a fixed parameter, the problem is PSPACE-complete.*

Proof. Let n be the number of agents in $\mathcal{IS}$, and let G be the set of global states in the model of length at most k. We take an arbitrary global state $h \in G$, such that $h = (h_e, h_1, ..., h_n)$, where h_i denotes the local state of agent $i \in \{1, .., n, e\}$.With a memory bound k, each local state of agent i, h_i is composed of $k + 1$ single observations, i.e. agent i remembers the previous $k + 1$ observations.

We use $h_i = \langle l_i^m \dots l_i^{m+k} \rangle$ to express such property, where $l_i^j \in L_i$ denotes the single observation at time j for $j > 0$.

In the memoryless case, each local state of an agent is the same as a single state in the bounded recall semantics, and let $|L_i|$ be the number of single observations that agent i has. The state space H_i generated from L_i has a size of $|L_i|^{k+1}$.

Therefore the number of global states in the model is bounded by $O(\prod_{i=1}^{n} |L_i|^{k+1})$ since a global state needs to be composed of local states. In a general case where k is not a fixed number, we get an exponential result for the upper bound. If we omit the constant term and consider k as a fixed parameter, the complexity is in polynomial space.

The semantics of $\mathrm{CTLK_{BR}}$ can be checked by using the procedure *VERIFY*(φ, h), reported below, which operates recursively by the depth of the formula. The cases are as follows:

- *VERIFY*(p, h): Check whether or not p is true in h, return *YES* if it is true, *NO* otherwise.
- *VERIFY*$(\varphi \wedge \phi, h)$: First call *VERIFY*(φ, h). If it returns *NO*, return *NO*. Otherwise output the result of *VERIFY*(ϕ, h).
- *VERIFY*$(\varphi \vee \phi, h)$: First call *VERIFY*(φ, h). If it returns *YES*, then return *YES*. Otherwise output the result of *VERIFY*(ϕ, h).
- *VERIFY*$(EX\varphi, h)$: Iterate over the set of states $h' \in G$, and for each state h', check if it is reachable from h. If it is reachable from h, then call *VERIFY*(φ, h') checking if φ is satisfied in h'. If *VERIFY*(φ, h') returns *YES*, return *YES*. If no such state h' is found, then return *NO*. This means the procedure needs to store the value of the state h, and it uses a polynomial amount of space.
- *VERIFY*$(E(\varphi U \phi), h)$: Iterate over the set of states $h' \in G$, and for each state h', call *VERIFY*(ϕ, h'). If it returns *YES*, check if $h' = h$, and if it is, return *YES*. Otherwise, check if there is a sequence of states from h to h' where ϕ holds in every states along this path. This check can be done by calling another procedure *PATH*(h, h', ϕ, $log(|\mathcal{IS}|)$) which checks if there is such a valid path of length less than or equal to the size of the model $|\mathcal{IS}|$, as G includes all possible global states. It returns *YES* if there is. The procedure *PATH* can be found in (Papadimitriou 1994). This is essentially a reachability problem, with an extra check on whether the formula is satisfied in the state. There will be at most $|\varphi|$ checks, where $|\varphi|$ denotes the size of the input formula. Therefore, it will use at most $O(|\mathcal{IS}|^2 \times |\varphi|)$ polynomial space.
- *VERIFY*$(EG\varphi, h)$: Check if there exists a path of length $|\mathcal{IS}|$ such that *VERIFY*(φ, h') holds for all states along the path. This again can be seen as a reachability problem, and uses polynomial space.
- *VERIFY*$(K_i\varphi, h)$: Iterate through all states $h' \in G$, and for each state, check if $h' \sim_i^k h$. For all h' such that $h' \sim_i^k h$, call *VERIFY*(φ, h'), and return *NO* if *VERIFY*(φ, h') returns *NO*. If *VERIFY*(φ, h') returns *YES* for every h', then return *YES*.
- *VERIFY*$(E_A\varphi, h')$: Iterate over all states $h' \in G$, and if $h \sim_i^k h'$ for any $i \in A$, call *VERIFY*(φ, h'). Return *YES* if *VERIFY*(φ, h') for all $g \in G$ such that $g \sim_i^k g'$ for any $i \in A$. Otherwise return *NO*.
- *VERIFY*$(D_A\varphi, g')$: Iterate over all states $g' \in G$, and if $g \sim_i^k g'$ for all $i \in A$, call *VERIFY*(φ, g'). Only return *YES* if *VERIFY*(φ, g') for all $g \in G$ such that $g \sim_i^k g'$ for any $i \in A$. Otherwise return *NO*.

The depth of calls will be at most $|\varphi|$. Given that the complexity of model checking against CTLK has been shown to be PSPACE-complete (Lomuscio and Raimondi 2006) we can conclude that the complexity of model checking against CTLK under bounded recall is PSPACE-complete when the window size is a fixed parameter, with respect to the size of the model and the size of the CTLK formula. □

The model checking problem of CTLK under a memoryless setting with incomplete information has been shown to be PSPACE-complete, which is the same complexity as we have shown for bounded perfect recall. However, the same model checking problem under a perfect recall setting has a higher complexity result of NON-ELEMENTARY(Dima 2008).

Theorem 3 concerns symbolic model checking, i.e., the verification problem when the model is not given explicitly. If the Kripke model is given explicitly from an interpreted system where states and transition relations are specified explicitly, the model checking problem against CTLK specification under bounded recall is PSPACE. In the same scenario where k is fixed and the bounded histories are given explicitly, the model checking problem is PTIME-complete, as there is no need for model construction.This is in the same complexity class as the model checking problem under observational semantics; indeed, note that in this case the same labelling algorithm memoryless semantics can be applied to bounded recall semantics.

Implementation

We implemented the algorithms of the previous section into an experimental checker called MCMAS_{BR} (MCMAS_{BR} 2020), which extends the open source checker MCMAS (Lomuscio, Qu, and Raimondi 2017) by adding support to bounded recall semantics; functionality for observational semantics is retained. Agents in MCMAS_{BR} remember a fixed number of the latest states visited in the run. Therefore, the agents' decisions are based on bounded local histories, rather than on their present state as under observational semantics. When an agent makes a transition, the oldest local state is deleted and a fresh one is added as in a FIFO queue.

MCMAS_{BR} takes as input an ISPL file describing the MAS under analysis and a set of formulas to be verified. Verification under bounded recall semantics is carried out by invoking the tool with the command-line flag `-bpr [window_size]`, where the size of the recall window is specified by the user. For simplicity, in the present version all agents have the same window size, which encodes the actual number of local states that agents remember.

```
History:
(state[0]=wait) and (state[1]=wait): {away};
end History
...
Evaluation
 waited2 if (Train1.state[0]=wait)
        and (Train1.state[1]=wait);
end Evaluation
```

Figure 1: BR referencing in ISPL.

To support the bounded semantics previously introduced, we modified the original ISPL syntax to allow users to specify protocols also on the basis of past observations as prescribed by Definition 1, as well as assignment functions. This allows atomic propositions to be assigned to finite histories instead of single observations. The revised expressivity supports a number of features including functionality for referring precisely to some of the past local states. As an example of this, see Figure 1, where the agent only waits for two timestamps before they leave. We refer to the sources for more details and examples[0].

Upon invocation, the tool parses the input ISPL file and then constructs the model as defined in Definition 1, where the agents' local states have their dimension fixed by the window length defined by the user. The model construction phase generates the set of bounded histories which are of arbitrary lengths up to the window size. MCMAS$_{BR}$ implements several methods to minimise the memory and computational overheads generated by the bounded recall semantics. For example, since at each time stamp in a run, the agents' local states need to evolve as described in Section 2, the symbolic encoding of each local history contains the composition of the previous history with the new variable assignments representing portions of the local history. This optimises the BDD memory used for computing and storing large local histories, notably during the subset construction stage where Boolean variables are generated to encode the state space. Agents' protocols were also modified and optimised to account for the BR semantics, and are defined on the basis of local histories also enabling reference to previous observations.

Evaluation. Intuitively, the increased expressivity of the bounded recall semantics comes at a cost of a larger number of Boolean variables required when compared against the standard observational semantics. This is expected to cause a performance degradation in the verification step. Note, however, that this is still preferable to the NON-ELEMENTARY complexity of perfect recall. To evaluate the cost of bounded recall, we now report how the performance of MCMAS$_{BR}$ scales as we increase the value of the window size and the example size.

N	W	$\mathbf{S}_R$	$\mathbf{S}_P$	t_m (s)	$t_v(s)$	**B (MB)**
1	0	9	16	0.001	0.001	8.98
	1	37	136	0.012	0.001	9.18
	2	149	560	0.039	0.002	9.92
	3	597	2380	0.551	0.017	11.75
	4	2389	6748	10.458	0.325	13.00
2	0	12	48	0.003	0.001	8.99
	1	26	1176	0.016	0.001	9.15
	2	62	18472	0.065	0.006	10.03
	3	134	2.13×10^5	1.069	0.116	11.65
	4	2389	1.93×10^6	19.073	2.599	14.50
3	0	24	96	0.004	0.001	8.99
	1	51	4656	0.031	0.002	9.26
	2	123	1.48×10^5	0.277	0.065	10.82
	3	267	3.47×10^6	10.385	66.35	13.37
	4	555	6.46×10^7	503.584	163.513	32.09
4	0	30	160	0.005	0.001	9.03
	1	63	1.29×10^4	0.123	0.004	9.37
	2	153	6.83×10^5	66.791	187.058	11.52
	3	333	2.63×10^7	114.8	37.364	27.97
	4	-	-	$\geq 5.0 \times 10^3$	-	-

Table 1: Experimental results for the *NTP*.

To do so we report the experiments obtained by implementing a *N-Transmission-Protocol* example in ISPL, which is similar to the well known *Bit-Transmission-Protocol* (Fagin et al. 1995). In *NTP*, the sender sends packets which contain random numbers in the set $\{1, ..., N\}$ over an unreliable channel. The receiver only responds with an ack when it receives N packets with unique numbers in N steps consecutively. Such system requires the agents to remember the previous packets they received. The CTLK specification checked for the system was:

$$AG(\bigwedge_{i=1}^{N} seen_i \rightarrow EFrec\_ack)$$

Here rec_ack represents the atomic proposition assigned to states when the sender receives an acknowledgement from the receiver. The atomic proposition $seen_i$ is assigned to histories where the agent has seen the digit i in the previous N timestamps. This expresses the property that if the receiver receives N unique set of digits consecutively, the sender will receive an acknowledgement from the receiver at some point. Intuitively, if the window size is less than N, then the atomic propositions do not exist, and the formula is therefore not meaningful, but it holds otherwise.

Table 1 shows the experimental results obtained with MCMAS$_{BR}$ running on an Intel® Core™ i7-2600 CPU 3.40GHz machine with 16GB RAM running Ubuntu v18.04.2 (Linux kernel v4.15). We record model construction time (t_m) and verification time (t_v) separately. This is because there is a model construction stage after parsing the ISPL file, where the model is built under bounded recall semantics and the computational time for reachability will take longer. The model construction time increases exponentially with the window size (W); this is mostly due to the generation of a larger state space. In the table S$_R$ represents the set of reachable states, S$_P$ represents the set of possible states of the model, B the BBD memory memory used.

To further evaluate the scalability of MCMAS$_{BR}$, we now report the experimental results obtained in Table 2 when verifying the TGC example against the specification ϕ discussed in Section 2, where the example is here extended to contain 10 trains and 8 trains. The assignment function for

N	W	$\mathbf{S}_R$	$\mathbf{S}_P$	t_m (s)	$t_v(s)$	**B (MB)**
8	0	0.83×10^2	13,122	0.037	0.002	9.46
	1	0.31×10^5	8.61×10^7	0.263	0.011	11.04
	2	2.50×10^6	3.76×10^{11}	0.888	0.113	16.6
	3	3.27×10^7	1.23×10^{15}	8.654	0.41	33.34
10	0	3.84×10^3	1.18×10^5	0.194	0.003	9.71
	1	3.35×10^5	6.97×10^9	0.571	0.015	12.0
	2	7.80×10^7	2.75×10^{14}	1.918	0.226	31.94
	3	1.93×10^9	8.10×10^{18}	13.14	0.576	36.55

Table 2: Experimental results for the *Train Gate Controller* with N trains.

atomic propositions differs for memoryless and bounded recall semantics, depending on the window size. For example, as we discussed earlier in Section 2, the atomic proposition $waited_i\_3s$ does not hold in any state when the window size is smaller than 2. We refer to the source of the model in ISPL format for more details^0.

In general, we observe from both examples that the model construction time increases more significantly than the verification time. This is to be expected due to an increase in the resulting BDD variables.The results validate the correctness of the implementation and the increased functionality obtained over observational semantics. The degradation of the performance is significant; it is however expected and also needs to be compared against the high complexity of the perfect recall case.

In summary, while the tool's performance degrades when increasing the window size, the results demonstrate that the method provides a concrete way for verifying an approximation of perfect recall.

Conclusions and Future Work

There is much literature on verification approaches for MAS against temporal-epistemic specifications (Penczek and Lomuscio 2003; Gammie and van der Meyden 2004). Most approaches support observational semantics. MCK is the only tool supporting perfect recall, but this is limited to one agent only, since the problem is undecidable for MAS of 2 agents or above. For this reasons, it is of interest to establish methods that can deal with approximations.

In this paper, we have pursued this direction by developing a bounded recall semantics and a fully implemented verification method for it. Intuitively, bounded recall sits in between observational semantics and full perfect recall. The size of the recall, defined on the length of local history which is remembered, determines the recall capabilities of the agent. Perfect recall can be seen as bounded recall with infinite window size; bounded perfect recall collapses to observational semantics when the window size is equal to 0.

To evaluate the approach we have studied the verification problem against implicitly given models and showed that model checking is in EXPTIME. This is a worse complexity than observational semantics but better than perfect recall, which is undecidable. Moreover when fixing the window size, the problem becomes PSPACE, which is the same as the complexity under observational semantics.

We implemented bounded recall on top of the open source model checker MCMAS paying particular attention to optimising the BDD structures resulting from the local histories. The experimental results that we obtained confirm that the performance degrades with higher recall sizes, but is manageable in some key benchmarks for sizes up to around 10.

In the future we intend to explore avenues to improve the scalability of the toolkit. For example, parallel computation can be potentially considered to speed up the construction of the state-space. Also it may be of interest to establish procedures whereby full recall is computed as the fix point of iterations on bounded recall. We leave this for further work.

Acknowledgments

A. Lomuscio is supported by a Royal Academy of Engineering Chair in Emerging Technologies. E. Yu acknowledges the support of the Austrian FWF grant W1255-N23 and the LIT AI Lab funded by the State of Upper Austria. F. Belardinelli acknowledges support from the ANR JCJC Project SVeDaS (ANR-16-CE40-0021).

References

Ågotnes, T., and Walther, D. 2009. A logic of strategic ability under bounded memory. *Journal of Logic, Language and Information* 18(1):55–77.

Alur, R.; Henzinger, T. A.; and Kupferman, O. 1997. Alternating-time temporal logic. In *FOCS*, 100–109. IEEE Computer Society.

Baier, C., and Katoen, J. 2008. *Principles of model checking*. MIT Press.

Belardinelli, F.; Lomuscio, A.; and Malvone, V. 2018. Approximating perfect recall when model checking strategic abilities. In *KR*, 435–444. AAAI Press.

Belardinelli, F.; Lomuscio, A.; and Malvone, V. 2019. An abstraction-based method for verifying strategic properties in multi-agent systems with imperfect information. In *Proc. AAAI*.

Biere, A.; Cimatti, A.; Clarke, E. M.; Fujita, M.; and Zhu, Y. 1999. Symbolic model checking using SAT procedures instead of BDDs. In *DAC*, 317–320. ACM Press.

Bryant, R. E. 1986. Graph-based algorithms for Boolean function manipulation. *IEEE Trans. Computers* 35(8):677–691.

Clarke, E. M., and Grumberg, O. 1987. Avoiding the state explosion problem in temporal logic model checking. In *PODC*, 294–303. ACM.

Clarke, E. M.; Henzinger, T. A.; Veith, H.; and Bloem, R., eds. 2018. *Handbook of Model Checking*. Springer.

Denning, P. J.; Comer, D.; Gries, D.; Mulder, M. C.; Tucker, A. B.; Turner, A. J.; and Young, P. R. 1989. Computing as a discipline. *IEEE Computer* 22(2):63–70.

Dima, C. 2008. Revisiting satisfiability and model-checking for CTLK with synchrony and perfect recall. In *CLIMA*, volume 5405 of *Lecture Notes in Computer Science*, 117–131. Springer.

Fagin, R.; Halpern, J.; Moses, Y.; and Vardi, M. 1995. *Reasoning About Knowledge*. The MIT Press.

Fagin, R.; Halpern, J. Y.; Moses, Y.; and Vardi, M. Y. 1997. Knowledge-based programs. *Distributed Computing* 10(4):199–225.

Gammie, P., and van der Meyden, R. 2004. MCK: model checking the logic of knowledge. In *CAV*, volume 3114 of *Lecture Notes in Computer Science*, 479–483. Springer.

Halpern, J. Y., and Vardi, M. Y. 1986. The complexity of reasoning about knowledge and time: Extended abstract. In *STOC*, 304–315. ACM.

Halpern, J. Y., and Vardi, M. Y. 1989. The complexity of reasoning about knowledge and time. I. lower bounds. *J. Comput. Syst. Sci.* 38(1):195–237.

Huang, X.; Chen, Q.; and Su, K. 2015. The complexity of model checking succinct multiagent systems. In *IJCAI*, 1076–1082. AAAI Press.

Jamroga, W.; Knapik, M.; and Kurpiewski, D. 2017. Fixpoint approximation of strategic abilities under imperfect information. In *AAMAS*, 1241–1249. ACM.

Lomuscio, A., and Penczek, W. 2007. Symbolic model checking for temporal-epistemic logics. *SIGACT News* 38(3):77–99.

Lomuscio, A., and Raimondi, F. 2006. The complexity of model checking concurrent programs against CTLK specifications. In *AAMAS*, 548–550. ACM.

Lomuscio, A.; Qu, H.; and Raimondi, F. 2017. MCMAS: an open-source model checker for the verification of multi-agent systems. *STTT* 19(1):9–30.

Lomuscio, A.; Qu, H.; and Russo, F. 2010. Automatic data-abstraction in model checking multi-agent systems. In *MoChArt*, volume 6572 of *Lecture Notes in Computer Science*, 52–68. Springer.

MCMAS$_{BR}$. 2020. MCMAS$_{BR}$.
https://vas.doc.ic.ac.uk/software/mcmas/extensions/.

Meski, A.; Penczek, W.; Szreter, M.; Wozna-Szczesniak, B.; and Zbrzezny, A. 2014. BDD-versus SAT-based bounded model checking for the existential fragment of linear temporal logic with knowledge: algorithms and their performance. *Autonomous Agents and Multi-Agent Systems* 28(4):558–604.

Mogavero, F.; Murano, A.; and Vardi, M. Y. 2010. Reasoning about strategies. In *FSTTCS*, volume 8 of *LIPIcs*, 133–144. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

Papadimitriou, C. H. 1994. *Computational complexity*. Addison-Wesley.

Penczek, W., and Lomuscio, A. 2003. Verifying epistemic properties of multi-agent systems via bounded model checking. *Fundam. Inform.* 55(2):167–185.

van der Hoek, W., and Wooldridge, M. J. 2002. Tractable multiagent planning for epistemic goals. In *AAMAS*, 1167–1174. ACM.

van der Meyden, R., and Shilov, N. V. 1999. Model checking knowledge and time in systems with perfect recall (extended abstract). In *FSTTCS*, volume 1738 of *Lecture Notes in Computer Science*, 432–445. Springer.

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