Model Checking Probabilistic Knowledge: A PSPACE Case

Xiaowei Huang and Marta Kwiatkowska
University of Oxford, UK

Abstract

Model checking probabilistic knowledge of memoryful semantics is undecidable, even for a simple formula concerning the reachability of probabilistic knowledge of a single agent. This result suggests that the usual approach of tackling undecidable model checking problems, by finding syntactic restrictions over the logic language, may not suffice. In this paper, we propose to work with an additional restriction that agent's knowledge concerns a special class of atomic propositions. A PSPACE-complete case is identified with this additional restriction, for a logic language combining LTL with limit-sure knowledge of a single agent.

Introduction

For a system with multiple agents, the properties concerning agents' knowledge can be important for its correctness. For instance, for a system with a diagnosis component, it is essential to have the property that, when a failure of the system occurs, the component can eventually *know* it. Some specific action (e.g., recover or terminate) can then be taken once the component gains the knowledge. Temporal epistemic logic (Fagin et al. 1995) has been developed to express these properties.

When considering stochasticity in the environment, e.g., imprecise sensor information, component failure or the occurrence of random events, it is necessary to formalise probabilistic knowledge. To work with such stochastic multiagent systems, the knowledge modality needs to be endowed with probabilistic measures, to express properties such as "once a failure occurs, the diagnosis component can eventually know this with probability greater than 90%".

However, it has been shown in (Huang, Su, and Zhang 2012; Huang 2013) that the model checking problem for a logic of probabilistic knowledge and discrete time is undecidable: the emptiness problem of probabilistic finite automata (Paz 1971) can be reduced to the model checking of a very simple formula concerning the reachability of a single agent's probabilistic knowledge over an atomic proposition. Moreover, due to the undecidability of value 1 problem of probabilistic finite automata (Gimbert and Oualhadj 2010) it is a reasonable conjecture that model checking *limit-sure knowledge* is also undecidable. Intuitively, we say that agent i has limit-sure knowledge over ϕ to mean that i knows ϕ with probability arbitrarily close to 1. These pessimistic results suggest that the usual approach of achieving decidability for an undecidable model checking problem, by finding syntactic restrictions over the logic language, may not suffice in this particular case.

The goal of this paper is to identify a sufficiently general decidable logic fragment for partial information stochastic multiagent systems that admits efficient model checking. More specifically, we suggest working with formulas in which the agent's limit-sure knowledge concerns a special class of atomic propositions, named resource propositions to denote that they have the same value in all initial states and can change their value at most once in a run. These propositions have been the subject of many studies, e.g., diagnosability of discrete-event systems and winning of multiplayer games, and can express a wide range of properties such as safety, reachability and diagnosability. With this additional restriction, we obtain a logic combining LTL and an agent's limit-sure knowledge with PSPACE-complete model checking complexity, same as LTL model checking.

Preliminaries: Stochastic MAS and PLTLK

A multiagent system (MAS) consists of a set of agents running simultaneously within an environment (Fagin et al. 1995). At each time point, every agent is in some *local state*, and the environment is in some *environment state*. A global state is formed from an environment state and local states, one for each agent. At a global state, every agent will make an observation over the system, take a *local action* and update its local state, and the environment will update the environment state according to the joint local action of the agents.

A stochastic multiagent system (SMAS) (Huang, Su, and Zhang 2012; Huang 2013) introduces probabilistic characteristics into a multiagent system: initial probability and transition probability. Initial probability assumes a probabilistic distribution over the states. Transition probability assumes a probabilistic distribution over the next possible environment states after considering the joint action from the agents. Moreover, (Huang, Luo, and van der Meyden 2011) defines a stochastic multiagent system in which agents take

randomised protocol. Formally, a finite SMAS is a tuple $M = (Agt, Prop, S, \{Act_i\}_{i\in Agt}, \{N_i\}_{i\in Agt}, \{O_i\}_{i\in Agt}, PI, PT, \pi)$, where Agt is a finite set of agents, $Prop$ is a finite set of atomic propositions, S is a finite set of environment states, Act_i is a finite set of local actions of agent $i \in Agt$ such that $Act = Act_1 \times ... \times Act_n$ is a set of joint actions, $N_i : S \to \mathcal{P}(Act_i) \setminus \{\emptyset\}$ provides on every state a nonempty set of local actions that are available to agent i, $PI : S \to [0..1]$ is an initial probability distribution such that $\sum_{s\in S} PI(s) = 1$, $PT : S \times Act \times S \to [0, 1]$ is a probabilistic transition function such that $\sum_{s'\in S} PT(s, a, s') = 1$ for all $s \in S$ and $a \in Act$, $O_i : S \to O$ is an observation function for each agent $i \in Agt$ such that O is a set of possible observations, and $\pi : S \to \mathcal{P}(Prop)$ is an interpretation of the atomic propositions $Prop$ at the states. We require that, for all states $s_1, s_2 \in S$ and $i \in Agt$, $O_i(s_1) = O_i(s_2)$ implies $N_i(s_1) = N_i(s_2)$: an agent can distinguish two states with different sets of next available actions. A state s is an initial state if $PI(s) > 0$. Note that the set of states S is the set of environment states rather than the set of global states. Agent i's local states are derived from the observation function O_i.

To specify the properties of an SMAS, we have a logic PLTLK that combines temporal operators, knowledge operator, and probability measures. Its syntax is given by

$$\phi \ ::= \ p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid X\phi \mid \phi_1 \, U \, \phi_2 \mid K_i\phi \mid \mathtt{E}_i^{\bowtie h}\phi \mid \mathtt{A}_i^{\bowtie h}\phi$$

where $p \in Prop$, $i \in Agt$, h is a rational constant in $[0, 1]$, and $\bowtie$ is a relation symbol in the set $\{\leq, <, >, \geq\}$. Intuitively, formula $X\phi$ expresses that ϕ holds in the next time instant, $\phi_1 \, U \, \phi_2$ expresses that ϕ_1 holds until ϕ_2 becomes true, $K_i\phi$ expresses that agent i knows ϕ, $\mathtt{E}_i^{\bowtie h}\phi$ expresses that there exists a resolution of nondeterminism such that agent i knows ϕ with probability in relation $\bowtie$ with constant h, and $\mathtt{A}_i^{\bowtie h}\phi$ expresses that for all resolutions of nondeterminism agent i knows ϕ with probability in relation $\bowtie$ with constant h. The concept of resolution of nondeterminism will be explained later. Other operators can be obtained in the usual way, e.g., $F\phi \equiv True \, U \, \phi$, $G\phi \equiv \neg F\neg\phi$, etc.

A probability space is a triple (W, F, μ) such that W is a set, $F \subseteq \mathcal{P}(W)$ is a set of *measurable* sets in $\mathcal{P}(W)$, closed under countable union and complementation, and $\mu : F \to [0, 1]$ is a *probability measure*, such that $\mu(W) = 1$ and $\mu(U \cup V) = \mu(U)+\mu(V)$ if $U \cap V = \emptyset$. As usual, we define the conditional probability $\mu(U|V) = \frac{\mu(U\cap V)}{\mu(V)}$ when $\mu(V) \neq 0$.

The semantics of the language PLTLK is based on a variant of interpreted system (Fagin et al. 1995), named probabilistic interpreted system (PIS). A PIS introduces into an interpreted system additional information: every transition relation is labelled with a probability.

A *path* ρ is a finite or infinite sequence of states $s_0 s_1 \ldots$ such that, for all $k \geq 0$, there exists some $a \in Act$ such that $PT(s_k, a, s_{k+1}) > 0$. Given a path $\rho = s_0 s_1 \ldots$, we use $s(\rho, m)$ to denote the state s_m, and $s(\rho, 0..m)$ to denote the sequence $s(\rho, 0)...s(\rho, m)$ of states. A path ρ is *initialized* if $PI(s(\rho, 0)) > 0$. For a finite path ρ, we write $last(\rho)$ for its last state. Without loss of generality, we assume that $Agt = \{1, ..., n\}$. Let L_i be the set of local states of agent $i \in Agt$. A *global* state s of an SMAS is an $(n + 1)$-tuple $(s_e, s_1, \ldots, s_n)$ such that $s_e \in S$ and $s_i \in L_i$ for all $i \in Agt$. At a global state, each agent independently takes some local action, which represents the decision it makes. A *joint action* of a multi-agent system in some global state is an n-tuple $a = (a_1, \ldots, a_n)$ such that $a_i \in Act_i$ for all $i \in Agt$. The environment responds to the joint action by updating its state according to the probabilistic distribution defined in PT.

From each initialized infinite path ρ, one may define a run in a PIS. Time is represented discretely by using natural numbers. A *run* is a function $r : \mathbb{N} \to S \times L_1 \times \ldots \times L_n$ from time to global states. A pair (r, m) consisting of a run r and time m is called a *point*, which may also be written as $r(m)$. If $r(m) = (s_e, s_1, \ldots, s_n)$ then we define $s_x(r, m) = s_x$ and $s_x(r, 0..m) = s_x(r, 0) \ldots s_x(r, m)$, for $x \in \{e\} \cup Agt$. Let a system $\mathcal{R}$ be a set of runs, and we call $\mathcal{R}\times\mathbb{N}$ the *set of points* of $\mathcal{R}$. Relative to a system $\mathcal{R}$, we define the set $\mathcal{K}_i(r, m) = \{(r', m') \in \mathcal{R}\times\mathbb{N} \mid s_i(r', m') = s_i(r, m)\}$ to be the set of points that are, for agent i, indistinguishable from the point (r, m).

We work with the memoryful assumption `mf`, i.e., agents have sufficient memory to remember all the past observations. Note that we interpret the states of the SMAS as states of the environment. Given an initialized infinite path ρ, we obtain a run $\rho^{\mathtt{mf}}$ by defining each point $(\rho^{\mathtt{mf}}, m)$ with $m \in \mathbb{N}$ as follows. The environment state at time m is $s_e(\rho^{\mathtt{mf}}, m) = s(\rho, m)$. The local state of agent i at time m is $s_i(\rho^{\mathtt{mf}}, m) = O_i(s(\rho, 0)) \cdot \ldots \cdot O_i(s(\rho, m))$, representing that the agent remembers all its observations.

In a system with both nondeterminism and probability such as an SMAS, we need to carefully handle the issue of *measurability*: runs of such a system are not measurable unless all the nondeterministic choices are resolved. The usual treatment, e.g., that of Halpern and Tuttle (Halpern and Tuttle 1993), introduces adversaries to resolve nondeterminism. The set of runs that are consistent with the behaviour of a given adversary consist of a measurable space. In an SMAS, we assume that the nondeterminism in the system is resolved by agents taking actions. A joint strategy $\sigma : S^* \to Act$ of the agents provides for every finite path a joint action such that $\sigma(\rho)_i \in N_i(last(\rho))$ for all $i \in Agt$. We write $Path^I(M, \sigma)$ for the set of initialized infinite paths in M which are consistent with the strategy σ. For every joint strategy σ, we define an associated *cell* c as follows. First, we define a subset of runs

$$\mathcal{R}_c = \{\rho^{\mathtt{mf}} \mid \rho \in Path^I(M, \sigma)\}.$$

We now define a probability space $(\mathcal{R}_c, F_c, \mu_c)$ using a well-known construction (e.g., that of (Vardi 1985)). Given a finite initialized path ρ of $m + 1$ states and m actions, write $\mathcal{R}_c(\rho) = \{r \in \mathcal{R}_c \mid s_e(r, 0..m) = s(\rho, 0..m)\}$ for the set of runs with prefix ρ. (One may view this as a *cone* of runs sharing the same prefix ρ.) Let F_c be the minimal algebra with basis the sets $\{\mathcal{R}_c(\rho) \mid \rho \text{ prefixes some } \rho' \in \mathcal{R}_c\}$, i.e., F_c is the set of all sets of runs that can be constructed from the basis by using countable union and complement. We define the measure μ_c on the basis sets by

$$\mu_c(\mathcal{R}_c(\rho)) = PI(s(\rho, 0))\times\prod_{i=0}^{m-1} PT(s(\rho, i), \sigma(s(\rho, 0..i)), s(\rho, i+1)).$$

There is a unique extension of μ_c that satisfies the constraints on probability measures (i.e., countable additivity and uni-

versality), and we also denote this extension by μ_c. For a joint strategy σ, we define a *cell* c as its corresponding probability space $(\mathcal{R}_c, F_c, \mu_c)$. A point (r,m) is in c if $r \in \mathcal{R}_c$. The set of indistinguishable points for agent i in (r,m) assuming c is $\mathcal{K}_i^c(r,m) = \mathcal{K}_i(r,m) \cap \{(r,m) \mid r \in \mathcal{R}_c, m \in \mathbb{N}\}$.

The probability space $(\mathcal{R}_c, F_c, \mu_c)$ provides a common prior over the set of runs in the cell c for all agents. Based on this, we can define a probability space for every agent i and every point (r,m) such that $r \in \mathcal{R}_c$. Let $\mathcal{R}(U) = \{r \in \mathcal{R} \mid \exists m : (r,m) \in U\}$ be the set of runs in $\mathcal{R}$ going through some point in the set $U \subseteq \mathcal{R} \times \mathbb{N}$. The probability information over c is $\mathbf{P}^c = \{PR_i^c \mid i \in Agt\}$, where PR_i^c is a function mapping each point (r,m) in c to a probability space $PR_i^c(r,m) = (\mathcal{R}(\mathcal{K}_i^c(r,m)), F_i^c(r,m), \mu_{r,m,i}^c)$ such that

1. $F_i^c(r,m) \subseteq \mathcal{P}(\mathcal{R}(\mathcal{K}_i^c(r,m)))$. Intuitively, at each point, each agent has a probability space in which the carrier is the set of runs going through points $\mathcal{K}_i^c(r,m)$.
2. The measure $\mu_{r,m,i}^c$ is defined by μ_c as

$$\mu_{r,m,i}^c(W) = \mu_c(W \mid \mathcal{R}(\mathcal{K}_i^c(r,m)))$$

where $W \in F_i^c(r,m)$ is a set of runs.

Given an SMAS M, a *probabilistic interpreted system* (PIS) is a tuple $\mathcal{I}(M) = (\mathcal{R}, C, \{\mathbf{P}^c\}_{c\in C}, \pi)$, where $\mathcal{R}$ is a system of runs, C is a set of cells in $\mathcal{R}$ such that $\mathcal{R} = \bigcup\{\mathcal{R}_c \mid c \in C\}$, $\{\mathbf{P}^c\}_{c\in C}$ is a set defining probability information for cells in C, and $\pi : \mathcal{R} \times \mathbb{N} \to \mathcal{P}(Prop)$ is an interpretation such that $\pi(r,m) = \pi(s_e(r,m))$.

The semantics of the language PLTLK in a PIS $\mathcal{I} = (\mathcal{R}, C, \{\mathbf{P}^c\}_{c\in C}, \pi)$ is given by interpreting formulas ϕ at points (r,m) of $\mathcal{I}$, using a satisfaction relation $\mathcal{I}, (r,m) \models \phi$, which is defined inductively as follows.

- $\mathcal{I}, (r,m) \models p$ if $p \in \pi(r,m)$,
- $\mathcal{I}, (r,m) \models \neg\phi$ if not $\mathcal{I}, (r,m) \models \phi$
- $\mathcal{I}, (r,m) \models \phi \wedge \psi$ if $\mathcal{I}, (r,m) \models \phi$ and $\mathcal{I}, (r,m) \models \psi$
- $\mathcal{I}, (r,m) \models X\phi$ if $\mathcal{I}, (r,m+1) \models \phi$.
- $\mathcal{I}, (r,m) \models \phi\, U\, \psi$ if there exists a time $m' \geq m$ such that $\mathcal{I}, (r,m') \models \psi$ and $\mathcal{I}, (r,m'') \models \phi$ for all m'' with $m \leq m'' < m'$
- $\mathcal{I}, (r,m) \models K_i\phi$ if for all points $(r',m') \in \mathcal{K}_i(r,m)$ we have that $\mathcal{I}, (r',m') \models \phi$.
- $\mathcal{I}, (r,m) \models \mathtt{A}_i^{\bowtie h}\phi$ if for all cells $c \in C$, either $\mathcal{K}_i^c(r,m) = \emptyset$, or $\mu_{r,m,i}^c(\mathcal{R}(\{(r',m') \mid \mathcal{I}, (r',m') \models \phi\})) \bowtie h$.

  Intuitively, the agent i has a certain probabilistic knowledge if, for all cells that the agent i thinks possible, the conditional probability of satisfying the property ϕ given the indistinguishability of agent i satisfies the specified relation $\bowtie$.
- $\mathcal{I}, (r,m) \models \mathtt{E}_i^{\bowtie h}\phi$ if there exists some cell $c \in C$ such that $\mathcal{K}_i^c(r,m) \neq \emptyset$ and $\mu_{r,m,i}^c(\mathcal{R}(\{(r',m') \mid \mathcal{I}, (r',m') \models \phi\})) \bowtie h$.

We will be interested in the problem of model checking formulas in this system. A formula ϕ is said to hold on M, written as $M \models \phi$, if $\mathcal{I}(M), (r,0) \models \phi$ for all $r \in \mathcal{R}$. The model checking problem is then to determine, given an SMAS M and a formula ϕ, whether $M \models \phi$.

We write $\overline{<}$ for $>$, $\overline{>}$ for $<$, $\overline{\leq}$ for $\geq$, and $\overline{\geq}$ for $\leq$, and write $\widehat{<}$ for $\leq$, $\widehat{>}$ for $\geq$, $\widehat{\leq}$ for $<$, and $\widehat{\geq}$ for $>$.

Proposition 1 *We have the following deduction rules for probabilistic knowledge operators.*

1. $\mathcal{I}, (r,m) \models \mathtt{A}_i^{\bowtie h}\phi$ *if and only if* $\mathcal{I}, (r,m) \models \mathtt{A}_i^{\overline{\bowtie} 1-h}\neg\phi$.
2. $\mathcal{I}, (r,m) \models \mathtt{E}_i^{\bowtie h}\phi$ *if and only if* $\mathcal{I}, (r,m) \models \mathtt{E}_i^{\overline{\bowtie} 1-h}\neg\phi$.
3. $\mathcal{I}, (r,m) \models \neg\mathtt{A}_i^{\bowtie h}\phi$ *if and only if* $\mathcal{I}, (r,m) \models \mathtt{E}_i^{\widehat{\bowtie} 1-h}\neg\phi$.
4. $\mathcal{I}, (r,m) \models \neg\mathtt{E}_i^{\bowtie h}\phi$ *if and only if* $\mathcal{I}, (r,m) \models \mathtt{A}_i^{\widehat{\bowtie} 1-h}\neg\phi$

It has been shown in (Huang, Su, and Zhang 2012; Huang 2013) that model checking PLTLK with memoryful semantics is undecidable, even for its single-agent fragment. Single-agent PLTLK is the fragment of PLTLK which allows only a fixed agent in a formula. Given a probabilistic finite automaton PA and a constant $\lambda \in [0,1]$, the emptiness problem (Paz 1971) is to determine the existence of a finite word such that it is accepted by PA with a probability greater than λ. The emptiness problem can be reduced to the model checking problem $M(PA) \not\models G\mathtt{A}_i^{\geq 1-\lambda}\neg p$ such that $M(PA)$ is obtained from PA by a PTIME construction and p is an atomic proposition.

In this paper, we identify a decidable fragment for this undecidable problem by placing restrictions on both the multi-agent system and the specification formula.

Almost-Sure and Limit-Sure Knowledge

The usual approach for tackling high complexity of a problem having a formula as one of its inputs is to find syntactic constraints on the logic language and show that the problem is computationally easier if the input formula satisfies the constraints. For temporal logic model checking, we work with CTL instead of CTL* whenever possible, since CTL model checking is PTIME-complete while CTL* model checking is PSPACE-complete. For temporal logic synthesis, the general problem is 2-EXPTIME complete (Pnueli and Rosner 1989), whereas there is a working fragment in polynomial time (Piterman, Pnueli, and Sa'ar 2006). For temporal epistemic model checking, the general problem is undecidable, with many decidable fragments obtained by reducing the expressiveness of the logic or the memory requirement of the agents (van der Meyden and Shilov 1999).

For probabilistic model checking, syntactic restrictions on logic languages can also be found by working with qualitative properties, see e.g., (Courcoubetis and Yannakakis 1995). In this paper, we write $\mathtt{A}_i^{\geq 1}\phi$ and $\mathtt{A}_i^{\leq 0}\phi$ for limit-sure knowledge, to express that agent i's knowledge about ϕ is arbitrarily close to 1 and 0, respectively. Given a system M and a formula ϕ with a set of limit-sure knowledge sub-formulas, we write $M \models \phi$ if, for any $\epsilon > 0$, there is $M \models \phi_\epsilon$, where ϕ_ϵ is obtained from ϕ by substituting limit-sure knowledge sub-formula $\mathtt{A}_i^{\geq 1}\phi$ and $\mathtt{A}_i^{\leq 0}\phi$ with $\mathtt{A}_i^{\geq 1-\epsilon}\phi$ and $\mathtt{A}_i^{\leq \epsilon}\phi$, respectively. Other than limit-sure knowledge, we may consider almost-sure knowledge (i.e., $\mathtt{A}_i^{=1}\phi$ or $\mathtt{A}_i^{=0}\phi$)[1]. The following theorem

[1]Please note that the almost-sure knowledge in (Huang 2013) is the limit-sure knowledge of this paper. We follow the definitions in (Chatterjee and Henzinger 2012) to differentiate almost-sure probability and limit-sure probability.

states that, for the problem we consider (i.e., agent's knowledge about atomic propositions), the almost-sure knowledge can be reduced to the usual nondeterministic knowledge.

Theorem 1 *Let $p \in Prop$ be an atomic proposition. Then we have that $\mathcal{I}, (r,m) \models \mathtt{A}_i^{=1} p$ if and only if $\mathcal{I}, (r,m) \models K_i p$, and $\mathcal{I}, (r,m) \models \mathtt{A}_i^{=0} p$ if and only if $\mathcal{I}, (r,m) \models K_i \neg p$.*

For the limit-sure knowledge, we notice that the value 1 problem of probabilistic finite automata, which is to determine the existence of a finite word such that it is accepted by *PA* with a probability arbitrarily close to 1, has been shown to be undecidable (Gimbert and Oualhadj 2010). It is therefore a reasonable conjecture that model checking limit-sure knowledge may also be undecidable in general.

These results suggest that, except for almost-sure knowledge, imposing restrictions over the input formulas may not be sufficient to obtain a decidable problem. In this paper, we propose to study the problem which concerns agents' limit-sure knowledge over a special class of atomic propositions.

Systems with Resource Propositions

An atomic proposition p is a resource proposition if it satisfies the following two conditions:

- for all states s_1, s_2 with $PI(s_1) > 0$ and $PI(s_2) > 0$, we have that $p \in \pi(s_1)$ if and only if $p \in \pi(s_2)$.
- for all runs $r \in \mathcal{R}$, there exists at most one time $m \in \mathbb{N}$ such that either $p \in s_e(r,m)$ and $p \notin s_e(r,m+1)$, or $p \notin s_e(r,m)$ and $p \in s_e(r,m+1)$.

Intuitively, a resource proposition has the same value on all initial states, and can change its value at most once on a run. As the name suggests, a resource proposition may be used to simulate a certain resource of the system that may or may not be consumed during a system execution. The resource, however, cannot be recovered once consumed.

The resource propositions are the subject of many studies and can express a wide range of properties. The results of the paper can therefore be applied to a large class of SMASs.

Example 1 *In a two-player game of reachability and safety objectives, the winning of an agent can be regarded as a resource proposition: for reachability, it is False at the initial states and turns into True when a goal state is reached; for safety, it is True at the initial states and turns into False whenever an unsafe state is reached.*

Example 2 *In the diagnosability of discrete-event systems, the occurrence of a failure event can be regarded as a resource proposition: it is False at the initial state and turns into True once a failure event occurs.*

In both examples, the value of a resource proposition can be changed at most once in a run. Let $KProp \subseteq Prop$ be a set of resource propositions. The checking of whether a given proposition p is a resource proposition can be done by standard LTL model checking. First of all, we transform the SMAS M into a nondeterministic system $U(M)$ by substituting probabilistic distributions with nondeterministic choices. The construction of $U(M)$ can follow the approach of constructing $N_i(M, \emptyset)$, which will be given in the next section. Then, we have the following conclusion.

Theorem 2 *A given proposition $p \in Prop$ is a resource proposition if and only if one of the following conditions holds:*

- $U(M) \models p \wedge ((p \, U \, G\neg p) \vee Gp)$
- $U(M) \models \neg p \wedge ((\neg p \, U \, Gp) \vee G\neg p)$

Although a resource proposition may start with either False or True, we only discuss the case where it starts with False. The other case can be obtained by introducing a new proposition which takes the negated value of the current proposition in all states.

Language with Restrictions

The following language is a fragment of the PLTLK language, named LTL(PK_i^{ls}) in the paper,

$$\begin{array}{lll} \phi & ::= & p \mid \neg p \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid X\phi \mid \phi_1 \, U \, \phi_2 \mid \psi \\ \psi & ::= & \mathtt{A}_i^{\geq 1} q \mid \mathtt{A}_i^{\leq 0} q \mid \psi_1 \vee \psi_2 \mid \psi_1 \wedge \psi_2 \end{array}$$

for $p \in Prop$ and $q \in KProp$. Intuitively, it imposes the following syntactic restrictions[2] on PLTLK: 1) temporal operators cannot appear in the scope of a probabilistic knowledge operator, 2) probabilistic knowledge operator can only express the limit-sure knowledge of a fixed agent i over resource propositions, and 3) the formulas are in negation normal form such that all probabilistic knowledge operators are positive (that is, $\mathtt{A}_i^{\bowtie h}\phi$ instead of $\mathtt{E}_i^{\bowtie h}\phi$). The expressiveness of the language LTL(PK_i^{ls}) subsumes the LTL language. Moreover, it should also be noted that, although the formula can only express a fixed agent's knowledge, the system may be composed of multiple agents.

Example 3 *For the diagnosability of a stochastic discrete-event system, the following LTL(PK_i^{ls}) formula expresses a diagnosability notion:*

$$G(p_f \Rightarrow F \, \mathtt{A}_{obsr}^{\geq 1} p_f) \tag{1}$$

which states that, once a failure event occurs, the observer obsr will eventually know this with probability arbitrarily close to 1.

The rule 1 of Proposition 1 enables the transformation between formulas of the form $\mathtt{A}_i^{\geq 1}\phi$ and $\mathtt{A}_i^{\leq 0}\phi$. Therefore, we assume in the following that all limit-sure knowledge formulas are of the form $\mathtt{A}_i^{\geq 1}\phi$, where ϕ is p or $\neg p$ for some resource proposition p.

In the following sections, we will show that model checking LTL(PK_i^{ls}) is PSPACE-complete. It seems that the problem is not harder than LTL model checking. However, we show that, while LTL model checking is NL-complete when the formula is fixed, model checking LTL(PK_i^{ls}) is PSPACE-complete when the formula is fixed.

[2]These constraints can also be seen as semantic restrictions because they consider limit-sure knowledge. For the pure syntactic restrictions of almost-sure knowledge, by Theorem 1, the model checking problem can be reduced to the problem of nondeterministic knowledge.

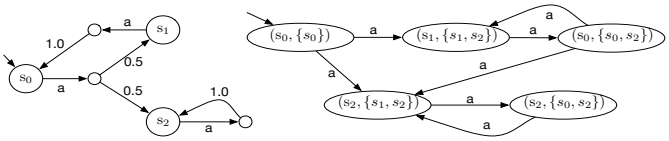

Figure 1: A stochastic multiagent system and its expanded knowledge-state system

PSPACE Upper Bound

Let ϕ be a formula and $\varphi = \neg\phi$ its negated formula. W.l.o.g., assume that φ is of negation normal form, which can be achieved by applying negation propagation rules of LTL formulas and the deduction rules of Proposition 1. Recall that $\neg \mathtt{A}_i^{\geq 1}\phi \equiv \mathtt{E}_i^{>0}\neg\phi$. All knowledge formulas in φ are of the form $\mathtt{E}_i^{>0}\phi$, for ϕ being p or $\neg p$ for some resource proposition p.

For every knowledge subformula $\mathtt{E}_i^{>0}\phi$ of φ, we introduce a new atomic proposition q^+, if $\phi = q$, or q^-, if $\phi = \neg q$. Let H_φ be the set of new atomic propositions of the formula φ. We use kq to range over propositions in H_φ.

Limit-Sure Knowledge Needs More Than A Subset Construction The usual approach (van der Meyden and Shilov 1999) of handling knowledge operators for a single agent proceeds by attaching to every state a set of states, called knowledge-state, representing the agent's knowledge at that state. Formally, a system $M = (S, I, \rightarrow, \pi)$ is expanded into a knowledge-state system $N(M) = (S \times \mathcal{P}(S), \{(s, P) \mid s \in I, P = \{t \in I \mid O_i(s) = O_i(t)\}\}, \rightarrow', \pi')$ such that $(s, P) \rightarrow' (t, Q)$ if $s \rightarrow t$ and $Q = \{t' \mid s' \in P, s' \rightarrow t', O_i(t') = O_i(t)\}$, and $\pi'((s, P)) = \pi(s)$. The model checking problem of memoryful semantics can then be reduced to the model checking problem of memoryless semantics. However, as we will show in the following example, this construction is insufficient when working with limit-sure knowledge.

Example 4 *Consider a system of a single agent i with 3 states $\{s_0, s_1, s_2\}$. The transition relation is given in the left diagram of Figure 1. The agent is blind, i.e., $O_i(s_0) = O_i(s_1) = O_i(s_2)$. A resource proposition p is such that $p \in \pi(s_2)$ but $p \notin \pi(s_0) \cup \pi(s_1)$. For this system, a specification formula $F\mathtt{A}_i^{\geq 1}p$ says that the agent i's knowledge about p will be arbitrarily close to* 1*. The right diagram of Figure 1 gives its expanded knowledge-state system. All knowledge-states of the expanded states contain one of the states in $\{s_0, s_1\}$. This means that $\neg p$ is always considered possible in these states, and therefore a wrong conclusion may be reached that $F\mathtt{A}_i^{\geq 1}p$ is unsatisfiable. However, we notice that $F\mathtt{A}_i^{\geq 1}p$ is satisfiable, because the probability of the runs satisfying $\neg p$ will be arbitrarily close to 0.*

In the next example, we present an approach for dealing with limit-sure knowledge by a 2-subset construction.

Example 5 *Consider a system of a single agent i with 4 states $\{s_0, s_1, s_2, s_3\}$. The transition relation is given in the top diagram of Figure 2. The agent is blind, i.e., $O_i(s_0) = O_i(s_1) = O_i(s_2) = O_i(s_3)$. A resource proposition p is such that $p \in \pi(s_1)$ but $p \notin \pi(s_0) \cup \pi(s_2) \cup \pi(s_3)$. For this system, a specification formula $F\mathtt{A}_i^{\geq 1}\neg p$ says that the agent i's knowledge about $\neg p$ will be arbitrarily close to* 1*.*

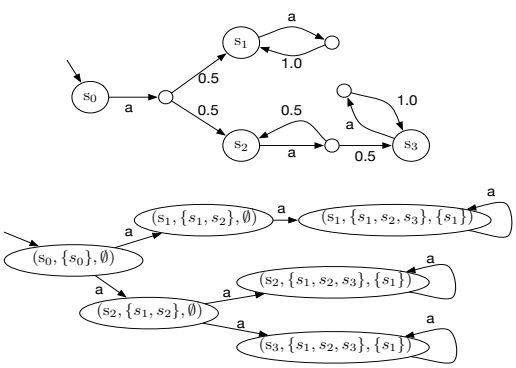

Figure 2: A stochastic multiagent system and its expanded probabilistic-knowledge-state system

Its knowledge-state system includes infinite paths in which the knowledge states in the loop contain the state s_1. According to Example 4, we cannot conclude that $F\mathtt{A}_i^{\geq 1}\neg p$ is unsatisfiable. The unsatisfiability can be justified with a 2-subset construction, as shown in the bottom of Figure 2. Each probabilistic-knowledge-state contains a state and two sets of states. The first two components are the same as those of knowledge-state systems. For the last component, it is a subset of the knowledge state, and once it is nonempty, it will be used to keep those states whose probability cannot be leaked by taking joint actions. See the algorithm for the explanation of how to prevent the probability leakage.

For this example, on any of the three loop states, we let $\{s_1\}$ be the subset of states in which the probability cannot be leaked. Therefore, we can conclude that $F\mathtt{A}_i^{\geq 1}\neg p$ is unsatisfiable. On the other hand, for the system in Example 4, there exists no subset of a knowledge state which can keep the probability. Therefore, the formula is satisfiable.

Every Limit-Sure Knowledge Needs A 2-Subset Construction For a formula with multiple knowledge operators of a single agent, the knowledge-state system is sufficient. However, for a formula with multiple limit-sure knowledge of a single agent, the probabilistic-knowledge-state system with a single 2-subset construction can be insufficient. In fact, for every limit-sure knowledge, a 2-subset construction is necessary. The reason is that different limit-sure knowledge may have counterexamples from different cells, which represent different joint strategies of the agents. Due to the space limit, we omit an example for this case.

Algorithm The above observations contribute to the following model checking algorithm. We describe the general idea first. As for the usual LTL model checking, we work with a product system of the probabilistic-knowledge-state system $N_i(M, H)$ and the formula automaton $\mathcal{A}_{\varphi_H}$. The algorithm proceeds through a depth-first search from an initial state s_0 to a state s_1, from which another depth-first search is performed to find a looping path back to s_1. The differ-

ence is that, for a new atomic proposition satisfiable on one of the loop states, its probability may be arbitrarily close to 0, see Example 4. So the looping path is required to have a second subset of states to record the probability. We use $H \subseteq H_\varphi$ to denote the subset of new atomic propositions that are required to be satisfiable on the looping path. Note that, for different counterexamples, there can be different H. In the nondeterministic algorithm, we will guess such a set and verify whether they are exactly those propositions satisfiable on the looping path.

Given a set $H \subseteq H_\varphi$, we can construct nondeterministic system $N_i(M, H)$ inductively as follows[3]. Let $N_i(M, \emptyset) = (\{i\}, Prop, S, Act_i, N_i, O_i, I, T, \pi)$ such that $I = \{s \mid PI(s) > 0\}$ is a set of initial states and, for any two states s, t and a joint action a, there is $(s, a, t) \in T$ if $PT(s, a, t) > 0$. Other components are the same as those of M.

Given $N_i(M, H') = (\{i\}, Prop', S', Act_i, N_i', O_i', I', T', \pi')$ and an atomic proposition $kq \in H \setminus H'$, we construct system $N_i(M, H' \cup \{kq\}) = (\{i\}, Prop'', S'', Act_i, N_i'', O_i'', I'', T'', \pi'')$ such that

- $Prop'' = Prop' \cup \{kq\}$, $S'' = S' \times \mathcal{P}(S) \times \mathcal{P}(S)$, $N_i''((s, P, Q)) = N_i'(s)$, $O_i''((s, P, Q)) = O_i'(s)$,
- $(s, P, Q) \in I''$ if $s \in I'$, $P = \{t \in S \mid PI(t) > 0, O_i(t) = O_i'(s)\}$, and $Q \subseteq P$,
- $((s_1, P_1, Q_1), a, (s_2, P_2, Q_2)) \in T''$ if $(s_1, a, s_2) \in T'$ and there exists a joint action $a' \in Act$ such that $P_2 = \{t \mid s \in P_1, PT(s, a', t) > 0, O_i(t) = O_i'(s_2)\}$, $Q_2 \subseteq P_2$, and $Q_2 = \{t \mid s \in Q_1, PT(s, a', t) > 0\}$ if $Q_1 \neq \emptyset$, and
- for all $p \in Prop'$, there is $p \in \pi''((s, P, Q))$ if $p \in \pi'(s)$; $kq \equiv q^+ \in \pi''((s, P, Q))$, if $\exists t \in P : q \in \pi(t)$, $\forall t \in Q : q \in \pi(t)$, and $Q \neq \emptyset$; and $kq \equiv q^- \in \pi''((s, P, Q))$, if $\exists t \in P : q \notin \pi(t)$, $\forall t \in Q : q \notin \pi(t)$, and $Q \neq \emptyset$.

Note that, in the transition relation, the set Q_2 contains all possible next states of Q_1 and is a subset of P_2. With this constraint, the probability does not leak. The property of resource propositions is useful to guarantee that, once a new atomic proposition kq holds on a state of a loop, it will be satisfiable on all the states of the loop.

For the formula $\varphi = \neg\phi$, we obtain φ' by substituting knowledge formulas $\mathrm{E}_i^{>0}\phi$ with their corresponding atomic propositions kq. Then we define

$$\varphi_H \equiv \varphi' \wedge \bigwedge_{kq \in H} GF\ kq \tag{2}$$

For those atomic propositions in H, we need to make sure that they are satisfiable infinitely often.

We can see that φ_H is a pure LTL formula. We can turn it into a Büchi automaton $\mathcal{A}_{\varphi_H} = (Q, \mathcal{P}(Prop), \delta, B, F)$ such that Q is a set of states, $\delta : Q \times \mathcal{P}(Prop) \to \mathcal{P}(Q)$ is a transition relation, $B \subseteq Q$ is a set of initial states, and $F \subseteq Q$ is a set of sets of acceptance states. Let $k = |H|$. Then from $N_i(M, H)$ and $\mathcal{A}_{\varphi_H}$, we construct a product system $N_i(M, H) \times \mathcal{A}_{\varphi_H} = (S'', I'', T'', \pi'')$ such that

- $S'' = S \times \Pi_{q \in H}(\mathcal{P}(S) \times \mathcal{P}(S)) \times Q$,
- $(s_0, P_1, Q_1, ..., P_k, Q_k, t) \in I''$ if $(s_0, P_1, Q_1, ..., P_k, Q_k) \in I'$ and $(t_0, \pi'((s_0, P_1, Q_1, ..., P_k, Q_k)), t) \in \delta$ for some $t_0 \in B$,
- $((s, P_1, ..., Q_k, t), (a_1, ..., a_k), (s', P_1', ..., Q_k', t')) \in T''$ if we have $((s, P_1, ..., Q_k), (a_1, ..., a_k), (s', P_1', ..., Q_k')) \in T'$ and $t' \in \delta(t, \pi'((s', P_1', ..., Q_k')))$, and
- $\pi''((s, P_1, ..., Q_k, t)) = \pi'((s, P_1, ..., Q_k))$.

Note that, in the above, to simplify the notation, we write a state $(((s_0, P_1, Q_1), ...), P_k, Q_k)$ of $N_i(M, H)$ as $(s_0, P_1, Q_1, ..., P_k, Q_k)$.

Now we have the following algorithm. Most details are omitted, including the collection of satisfiable new atomic propositions during the second depth-first search. The algorithm follows a similar pattern to that of collecting Büchi conditions held on the looping path, see e.g., (Gaiser and Schwoon 2009).

Theorem 3 *The checking of $M \models \phi$ can be done by nondeterministically guessing a set $H \subseteq H_\varphi$ and then verifying the following three conditions:*

1. *there exists an initial state s_0 in $N_i(M, H_\varphi) \times \mathcal{A}_{\varphi_H}$, from which a loop can be reached,*
2. *the loop satisfies all Büchi constraints, and*
3. *the loop requires exactly those atomic propositions in H to be satisfiable.*

For the complexity, we notice that the product system $N_i(M, H) \times \mathcal{A}_\varphi$ is of size exponential with respect to both the system M and the formula ϕ. The computation can be done on-the-fly by taking a polynomial size of space. Therefore, the complexity is in NPSPACE=PSPACE.

PSPACE Lower Bound

In this section, we show that PSPACE is the lower bound for program complexity of model checking LTL(PK_i^{ls}). We reduce a PSPACE-complete problem to model checking a fixed formula, which does not vary with the problem size.

Theorem 4 *The checking of $M \models \phi$ is PSPACE-hard for program complexity.*

We present the proof idea only. It is reduced from the problem of deciding if, for a given nondeterministic finite state automaton A over an alphabet Σ, the language $L(A)$ is equivalent to the universal language Σ^*. Let $A = (Q, q_0, \delta, F)$ be an NFA such that Q is a set of states, $q_0 \in Q$ is an initial state, $\sigma : Q \times \Sigma \to \mathcal{P}(Q)$ is a transition function, and $F \subseteq Q$ is a set of final states.

The proof proceeds by constructing a system $M(A)$ for a single agent i. $M(A)$ starts from an initial state s_0 and moves into two subsystems $M_1(A)$ and $M_2(A)$ by taking action *pass* and *test*, respectively. Intuitively, the subsystem $M_1(A)$ is to run any possible transition to simulate the universal language Σ^*, and the subsystem $M_2(A)$ is to simulate the automaton A. The initial states of the subsystems $M_1(A)$ and $M_2(A)$ are l_0 and (l_0, q_0), respectively, for some $l_0 \in \Sigma$.

In $M_1(A)$, from every state $l \in \Sigma$, one can reach a state (l', u) such that $l' \in \Sigma$ is any symbol and $u \notin \Sigma$ is a new symbol representing that the current state is an intermediate state from which two actions *pass* and *test* are available. Taking

[3] We describe it in an inductive way simply to ease notations.

any of the actions will lead to the state l'. In $M_2(A)$, from every state (l_1, q_1), one can reach a state (l_2, q_2^u) if $q_2 \in \delta(q_1, l_2)$. Similarly to $M_1(A)$, the state (l_2, q_2^u) is an intermediate state from which two actions *pass* and *test* are available. If the action *pass* is taken then the state (l_2, q_2) is reached. If the action *test* is taken then there are two cases depending on whether $l_2 \in F$ or not. If $l_2 \in F$ then both the state (l_0, q_0) and the state l_2 (in $M_1(A)$) are reached with probability 0.5. If $l_2 \notin F$ then the state (l_0, q_0) is reached.

The agent i can see and only see the symbols on the states, i.e., $O_i(l) = O_i((l, u)) = O_i((l, q)) = O_i((l, q^u)) = l$ for $l \in \Sigma$ and $q \in Q$. All states in $M_1(A)$ are labeled with proposition $finished$. We can show that the language of A is universal if and only if

$$M(A) \models F\mathtt{A}_i^{\geq 1} finished. \tag{3}$$

Related Work

Epistemic logic (Halpern and Moses 1990; Parikh and Ramanujam 1985) is a well known formalism for reasoning about *partial observation* multiagent systems, which provides means to state what agents *know*. When considering stochasticity in the environment (e.g., imprecise sensor information, the occurrence of random events, etc), it is necessary to formalise *probabilistic knowledge*. There has been little work to date on combining probabilistic and epistemic reasoning. This paper builds on the formalism of (Huang, Su, and Zhang 2012; Huang 2013) that defines probabilistic knowledge by quantifying over all possible adversaries, a typical approach employed to resolve nondeterminism for systems containing both nondeterminism and probability, as in e.g. (Halpern and Tuttle 1993).

Model checking and strategy synthesis for the related model of stochastic multiplayer games has been considered by several authors, e.g. (Chatterjee and Henzinger 2012; Chen et al. 2013; Basset et al. 2015), mainly focusing on multiobjective properties and the use of the strategy operator of ATL. However, all these papers reason about complete information systems, and therefore the corresponding logics do not include the knowledge operator. For partial observation games, existing work (Chatterjee and Doyen 2014) concerns memory requirement for a successful strategy to *qualitatively win* the game. It is unclear whether the verification of probabilistic knowledge can be directly reduced to the winning of a game.

Conclusions

The paper presents the first decidable logic fragment for probabilistic knowledge over partial information stochastic multiagent systems. The logic combines LTL with a single agent's limit-sure knowledge over resource propositions and its model checking is shown to be PSPACE-complete. In contrast to LTL model checking, the program complexity (by fixing the formula) is also PSPACE-complete.

Acknowledgments. The authors are supported by ERC Advanced Grant VERIWARE and EPSRC Mobile Autonomy Programme Grant EP/M019918/1. We also gratefully acknowledge the anonymous referees for their helpful comments.

References

Basset, N.; Kwiatkowska, M.; Topcu, U.; and Wiltsche, C. 2015. Strategy synthesis for stochastic games with multiple long-run objectives. In *TACAS 2015*, 256–271.

Chatterjee, K., and Doyen, L. 2014. Partial-observation stochastic games: How to win when belief fails. *ACM Transactions on Computational Logic* 15(2):16.

Chatterjee, K., and Henzinger, T. A. 2012. A survey of stochastic -regular games. *Journal of Computer and System Sciences* 78(2):394–413.

Chen, T.; Forejt, V.; Kwiatkowska, M.; Parker, D.; and Simaitis, A. 2013. Automatic verification of competitive stochastic systems. *Formal Methods in System Design* 43(1):61–92.

Courcoubetis, C., and Yannakakis, M. 1995. The complexity of probabilistic verification. *Journal of the ACM* 42(4):857–907.

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

Gaiser, A., and Schwoon, S. 2009. Comparison of algorithms for checking emptiness on buchi automata.

Gimbert, H., and Oualhadj, Y. 2010. Probabilistic automata on finite words: Decidable and undecidable problems. In *ICALP 2010*, 527–538.

Halpern, J. Y., and Moses, Y. 1990. Knowledge and Common Knowledge in a Distributed Environment. *JACM* 37(3):549–587.

Halpern, J. Y., and Tuttle, M. R. 1993. Knowledge, probability, and adversaries. *Journal of the ACM* 40(3):917–962.

Huang, X.; Luo, C.; and van der Meyden, R. 2011. Symbolic model checking of probabilistic knowledge. In *TARK XII*, 177–186.

Huang, X.; Su, K.; and Zhang, C. 2012. Probabilistic Alternating-time Temporal Logic of Incomplete information and Synchronous Perfect Recall. In *AAAI 2012*, 765–771.

Huang, X. 2013. Diagnosability in concurrent probabilistic systems. In *AAMAS 2013*, 853–860.

Parikh, R., and Ramanujam, R. 1985. Distributed Processes and the Logic of Knowledge. In *Logics of Programs 1985*, 256–268.

Paz, A. 1971. *Introduction to probabilistic automata (Computer science and applied mathematics)*. Academic Press.

Piterman, N.; Pnueli, A.; and Sa'ar, Y. 2006. Synthesis of reactive(1) designs. In *VMCAI 2006*, 364–380.

Pnueli, A., and Rosner, R. 1989. On the synthesis of a reactive module. In *POPL1989*, 179–190.

van der Meyden, R., and Shilov, N. V. 1999. Model Checking Knowledge and Time in Systems with Perfect Recall. In *FSTTCS 1999*, 432–445.

Vardi, M. Y. 1985. Automatic verification of probabilistic concurrent finite-state programs. In *FOCS 1985*, 327–338.