A Simple Integration of Epistemic Logic and Reinforcement Learning

Thorsten Engesser
TU Wien
Vienna, Austria
thorsten@logic.at

Thibaut Le Marre
ENS Rennes, ENS de Lyon, CNRS,
Université Claude Bernard Lyon 1,
Inria, LIP, UMR 5668, Lyon, France
thibaut.le-marre@ens-lyon.fr

Emiliano Lorini
IRIT, CNRS, Toulouse University
Toulouse, France
emiliano.lorini@irit.fr

François Schwarzentruber
ENS de Lyon, CNRS, Université
Claude Bernard Lyon 1, Inria,
LIP, UMR 5668, Lyon, France
francois.schwarzentruber@ens-lyon.fr

Bruno Zanuttini
Université Caen Normandie,
ENSICAEN, CNRS, Normandie Univ,
GREYC UMR6072, Caen, France
bruno.zanuttini@unicaen.fr

ABSTRACT

We propose an integration of epistemic logic with reinforcement learning via a semantics that uses the concept of belief bases. In our framework, an agent's subjective state is identified with their belief base, which captures the agent's personal representation of the environment. The agent's subjective state is distinguished from the global state, which captures the overall information about the environment and about the agent's belief base from an external perspective. We instantiate the concepts of global state and subjective state in Partially-Observable Markov Decision Process (POMDPs), defining so-called *Belief Base POMDPs (BB-POMDPs)*.

We show that in our epistemic framework, we can use the beliefs of the learning agent to formalize and implement a natural form of shielding, which prevents agents from performing actions that are not known to be safe. Our implementation of shielding relies on a model-checking algorithm to automatically verify whether a given fact is deducible from the agent's belief base.

We perform a case study of model-free reinforcement learning on a simple wumpus scenario, using a variant of Q-learning on the agent's subjective states, using the agent's beliefs for reward shaping and shielding. In particular, our experiments show that our version of shielding can successfully protect the agent from harm while improving the utility of the learned policy.

CCS CONCEPTS

• **Computing methodologies** → **Reinforcement learning**; **Partially-observable Markov decision processes**; **Reasoning about belief and knowledge**.

KEYWORDS

Epistemic logic; Reinforcement learning

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

ACM Reference Format:
Thorsten Engesser, Thibaut Le Marre, Emiliano Lorini, François Schwarzentruber, and Bruno Zanuttini. 2025. A Simple Integration of Epistemic Logic and Reinforcement Learning. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 9 pages.

1 INTRODUCTION

The need for an integration of machine learning (ML) and knowledge representation has been largely emphasized in the artificial intelligence (AI) community. According to Valiant [40], a key challenge for computer science is to come up with an integration of the two most fundamental phenomena of intelligence, namely, the ability to learn from experience and the ability to reason from what has been learned. The present paper is focused on the integration of epistemic logic and reinforcement learning (RL), with the aim of combining an agent's capacity to perform deductive reasoning with the capacity to learn the expected value of an action executed at a given state based on their past experiences.

Our integration is between partially-observable Markov decision processes (POMDPs) and epistemic logic. This is a natural move given their common focus on modeling an agent's uncertainty about the environment, while Markov decision processes (MDPs) suppose the environment is fully observable and the agent has no uncertainty about it. Our integration relies on a formal language for representing the learning agent's *explicit* and *implicit* beliefs. While explicit beliefs correspond to the information in the agent's belief base, implicit beliefs are all facts that the agent can deduce from their belief base.

Example 1. Consider a simplified version of the *Wumpus World* [42] on a grid of size $n \times n$ (see Figure 1). The agent starts at the bottom left corner of the grid. The goal is for the agent to reach the upper right corner. At each step, the agent has the following movement actions available: Up, Down, Right, Left, and Noop. It is not possible to move out of the grid: for example, if the agent is at the bottom of the grid, Down has the same effect as Noop.

The grid contains a fixed number of wumpuses, which is known to the agent. However, their locations are initially unknown. Wumpuses do not move and the agent knows that they are static. If the

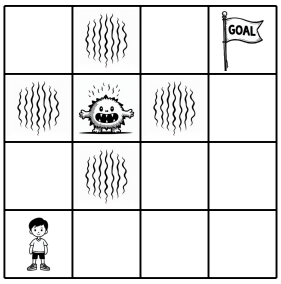

Figure 1: Our simplified Wumpus World. The agent in the bottom left corner has to reach the goal in the top right corner. They cannot see the wumpus, but they can smell the wumpus if they are standing on an adjacent cell.

agent moves into a cell occupied by a wumpus, the agent dies and the game is over. However, wumpuses have an offensive smell, which can be noticed at the cells adjacent to the wumpus. If the agent moves to such a cell, they get the explicit belief that there is a smell at that cell. If the agent moves to a cell that is not adjacent to a wumpus, they get the explicit belief that there is no smell at that cell. The agent can use these explicit beliefs to infer implicit beliefs, such as the belief that there is a wumpus (or that there is no wumpus) on a particular cell.

The language will be interpreted by means of a semantics which exploits belief bases. The idea of using belief bases as a formal semantics for representing epistemic attitudes of agents and for interpreting epistemic languages was developed in recent work by Lorini et al. [22, 24, 27–29]. The major advantage of this semantics, compared to the traditional semantics based on so-called multi-relational Kripke models [16], is its succinctness, which makes it well-suited for formal verification in real applications.

The first level of our integration will consist in adding information about the agents' explicit beliefs to the state description of POMDPs. This is a crucial step since an action of the agent could modify not only the environment but also their own beliefs. For example, an agent called Ann may perform the action of convincing another agent called Bob that it is sunny outside, which results in Ann explicitly believing that Bob explicitly believes that it is sunny outside. The second level of integration will consist in exploiting the agent's inferential capability offered by the epistemic logic for discarding those actions that, according to the agent's implicit beliefs (i.e., what the agent can infer), violate certain norms or constraints if they are executed in a given state. The agent will not need to learn the value of those actions, but they will simply exclude them from the action selection process. This action discarding mechanism is close to the notion of *shielding* [2]. In order to achieve the integration at the implementation level and not simply at a conceptual level, we will combine a Q-learning algorithm with a model checking algorithm for the epistemic language. We will show experimentally in a simple wumpus scenario that this integration of reinforcement learning with epistemic reasoning through model checking improves the agent's learning process, by making it obtain a higher utility in the long term.

The contributions of the paper are the following.

(1) We define a relevant model checking problem that can be used for RL. We identify a sufficiently expressive fragment of the logic of belief bases that makes model checking still efficient in practice. We show that our model checking problem is Θ_2-complete and give an efficient algorithm.
(2) We define the notion of *Belief Base POMDPs* (BB-POMDPs). They are an instantiation of standard POMDPs, and are defined with logical formulas.
(3) We conduct a case study, showing that *reward shaping* and *shielding* based on the belief of the learning agent (specified by logical formulas) can be useful in practice.

Impact for epistemic logic. Currently, the main use of epistemic logic is *epistemic planning*, where the agents plan the next actions taking into account mental states [10, 21, 38]. While epistemic planning is undecidable in general [4], fragments with a lower complexity have been identified [5, 6, 8, 13, 14, 30]. Reinforcement learning can be seen either as a new tool to cope with the complexity of epistemic planning, or as a way for the agent to do model-free learning.

Impact for RL. This paper offers a simple integration of an epistemic logic with both explicit and implicit beliefs, into reinforcement learning. It allows the modeling of environments where we distinguish what the agents explicitly believe and what they implicitly believe. Our approach also allows us to specify expert knowledge, which the agent can use to make inferences. As far as we know, this approach is not possible with traditional approaches in RL. Finally, we can leverage implicit beliefs to implement a form of *shielding* by model checking.

Outline. The paper is organized as follows. In Section 2 we discuss the related work. In Section 3 we recall the setting of belief bases given by Lorini [24]. In Section 4, we present the results about the model checking problem. In Section 5, we define so-called *Belief Base POMDPs*. In Section 6, we present the experimental results. In Section 7, we conclude.

2 RELATED WORK

Model Checking for POMDPs. Model checking in the context of POMDPs has traditionally focused on policy synthesis under probabilistic constraints. Bouton et al. [7] explored the use of Point-Based Value Iteration (PBVI) for maximizing the satisfaction of linear temporal logic (LTL) formulas in POMDPs. De Giacomo et al. [11] introduced the concept of restraining bolts as a method for enforcing constraints on reinforcement learning agents through external logical specifications. Unlike traditional RL settings, where constraints are directly incorporated into the environment or reward function, the restraining bolt operates separately, using LTLf (LTL on finite traces) formulas to guide the agent's learning process. The idea of restraining bolts is different from ours insofar as we disallow actions that do not comply with the epistemic logic specification, while they use a reward shaping mechanism to promote the compliance with the LTLf specification. Their approach was extended by Neufeld et al. [33] with deontic concepts.

Bayesian RL. Bayesian Reinforcement Learning provides a powerful framework for managing uncertainty by maintaining a probability distribution over hidden states. Bayesian RL methods are

often challenged by scalability and the complexity of integrating sophisticated knowledge into the learning process, as noted by Nguyen et al. [34]. In contrast, our framework simplifies the incorporation of expert knowledge through common ground rules. By allowing the agent to infer knowledge via logical deductions, our method makes expert knowledge integration straightforward, offering a tractable approach for real-world applications.

Epistemic MDPs. Epistemic Markov Decision Processes focus on information-gathering actions in static environments, where the agent's goal is to refine its belief state without altering the physical state of the world. Epistemic MDPs, as explored by Sabbadin et al. [36], are computationally demanding, with policy existence being PSPACE-complete for short-horizon problems.

Building on the idea of belief refinement, Araya-López et al. [3] introduced belief-dependent POMDPs (ρPOMDPs) as an extension of POMDPs where rewards depend on the agent's belief state. This extension allows for optimizing policies in environments where reducing uncertainty or gaining information is crucial.

While the philosophical foundations of our approach align with EMDPs and ρPOMDPs in terms of focusing on the agent's beliefs, our method avoids the complexity of maintaining probabilistic beliefs. Our beliefs are encoded in the states, allowing us to use model-free RL, which is not possible in EMDPs and ρPOMDPs. Still, we can specify rewards based on the agent's beliefs.

Deep RL. Recent advances in deep reinforcement learning for POMDPs often rely on memory-based architectures, such as Long Short-Term Memory (LSTM) networks, to retain past observations [17, 18, 31, 32]. In contrast, our approach uses the belief base constructed from the history of these interactions, effectively serving as a function of past experiences and observations.

3 BACKGROUND ON BELIEF BASES

In this section, we describe how to represent states that model both facts about the real world and the explicit beliefs of the agent. For this purpose, we rely on Lorini's belief base semantics for epistemic logic [24].[1]

3.1 States and Explicit Beliefs

We consider a countable set of propositions denoted by $Atm_0 = \{p, q, \ldots\}$ and a finite set of agents Agt. We then define the following language $\mathcal{L}_\triangle$ for explicit beliefs:

$$\alpha \quad ::= \quad p \mid \neg\alpha \mid \alpha \wedge \alpha \mid \triangle_i \alpha,$$

where p ranges over Atm_0, and i ranges over Agt. For all $\alpha \in \mathcal{L}_\triangle$, $i \in Agt$, the construction $\triangle_i\alpha$ is called an *epistemic atom* and is read 'agent i explicitly believes α'. The set of epistemic atoms is denoted by $EpAtm$. We define $Atm = Atm_0 \cup EpAtm$ to be the set of atoms (propositions and epistemic atoms). We denote by $Atm(\alpha)$ and $EpAtm(\alpha)$ all atoms and epistemic atoms occurring in some formula α.

Definition 2. A *state* is a valuation $\sigma \subseteq Atm$.

[1]We warn the reader that we use a different notation from the one used in [24]. In the original belief base semantics, the agents' belief bases were distinguished from the propositional valuation. In our current formulation, we use a single valuation that is used to interpret both propositions and epistemic atoms.

Example 3. In state $\sigma = \{p, r, \triangle_1 p, \triangle_2 q\}$, p is true, q is false, r is true, agent 1 explicitly believes p and agent 2 explicitly believes q, and there is no other explicit belief.

Assuming Atm is fixed, we denote the set of all states by Σ. Formulas of $\mathcal{L}_\triangle$ are interpreted relative to a state σ using the semantics of classical propositional logic:

$$\begin{aligned} \sigma &\models x \quad \text{if} \quad x \in \sigma \text{ for } x \in Atm, \\ \sigma &\models \neg\alpha \quad \text{if} \quad \sigma \not\models \alpha, \\ \sigma &\models \alpha_1 \wedge \alpha_2 \quad \text{if} \quad \sigma \models \alpha_1 \text{ and } \sigma \models \alpha_2. \end{aligned}$$

Definition 4. The *belief base* of agent i at a state σ is defined as

$$K_i(\sigma) = \{\alpha \in \mathcal{L}_\triangle \mid \triangle_i \alpha \in \sigma\}.$$

Example 5. Let $\sigma = \{p, \triangle_1 p, \triangle_1 q, \triangle_2 r\}$ be a state. This means that p is true, agent 1 explicitly believes p, explicitly believes q and agent 2 explicitly believes r. We have $K_1(\sigma) = \{p, q\}$. While the real state is σ, agent 1 sees $\{p, q\}$. That is, the agent believes p and falsely believes q. We have $K_2(\sigma) = \{r\}$.

3.2 Implicit beliefs

The agents' explicit beliefs induce a doxastic accessibility relation on the set of states. In particular, agent i considers state σ' possible at state σ (written $\sigma \mathcal{R}_i \sigma'$) if they have the same belief base in both σ and σ', and all formulas α believed by i in σ are true in σ'.

Definition 6. The *doxastic accessibility relation* for agent i is $\mathcal{R}_i \subseteq \Sigma \times \Sigma$ such that

$$\sigma \mathcal{R}_i \sigma' \text{ iff } K_i(\sigma) = K_i(\sigma') \text{ and } \forall \alpha \in K_i(\sigma), \sigma' \models \alpha.$$

Example 7. Let $\sigma = \{\triangle_i p\}$ and $\sigma' = \{\triangle_i p, p\}$. We have $\sigma \mathcal{R}_i \sigma'$ because $K_i(\sigma) = K_i(\sigma') = \{p\}$ and $p \in \sigma'$. We do not have $\sigma' \mathcal{R}_i \sigma$ because $p \notin \sigma$.

Following Lorini [22, 24], we can use this to extend the language $\mathcal{L}_\triangle$ with an implicit belief modality $\Box_i$, capturing what the agent can infer from the information in its belief base. The resulting language $\mathcal{L}_\Box$ is defined by the following grammar:

$$\varphi \quad ::= \quad \alpha \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Box_i \varphi,$$

where α ranges over $\mathcal{L}_\triangle$. The construction $\Box_i\varphi$ is read 'agent i implicitly believes φ'.

Formulas from $\mathcal{L}_\Box$ are interpreted with respect to a belief base σ and a set of belief bases U (which we call *context* or *universe*). Intuitively, U is the set of states that are compatible with the agents' *background information*. This corresponds to the notion of common ground in epistemic logic and linguistics [39]. Truth is defined inductively on formulas as follows (Boolean cases are omitted, since they are interpreted in the usual way):

$$U, \sigma \models \Box_i \varphi \quad \text{if} \quad \forall \sigma' \in U, \text{ if } \sigma \mathcal{R}_i \sigma' \text{ then } U, \sigma' \models \varphi.$$

Note that σ not necessarily belongs to U; when it does not, this means that the agents have wrong background information.

4 EFFICIENT MODEL CHECKING

The model checking problem for the multi-agent language of explicit and implicit belief is PSPACE-complete in general. The upper bound is proved in [12], while the lower bound is proved in [23]. We identify here two more tractable fragments. By more tractable

we mean Θ_2^P-complete, as stated in Proposition 15. We recall that Θ_2^P is the class of all decision problems which can be solved in deterministic polynomial time using a polynomial number of *independent* calls to an NP-oracle [41].[2]

4.1 Fragments

In the rest of the article, we consider the fragment defined by Davila et al. [10] in which $\Box_i$ cannot be nested. More precisely, we consider the fragment of formulas φ generated by:

$$\varphi ::= \alpha \mid \Box_i\alpha \mid \neg\varphi \mid \varphi \wedge \varphi$$
$$\alpha ::= p \mid \neg\alpha \mid \alpha \wedge \alpha \mid \triangle_i\alpha$$

where i ranges over Agt. We call this language $\mathcal{L}_{1\Box}$.

Example 8. Formula $p \wedge \Box_1 p \wedge \Box_2\triangle_1\neg p$ is in $\mathcal{L}_{1\Box}$, but $\Box_1\Box_1 p$ and $\Box_1\Box_2 p$ are not, because we disallow two $\Box_i$ to be nested.

We also introduce the *subjective fragment* for a fixed agent L, denoted by $\mathcal{L}_{1\Box}^{\mathrm{L}}$ (which is a fragment of $\mathcal{L}_{1\Box}$) by:

$$\varphi ::= \triangle_{\mathrm{L}}\alpha \mid \Box_{\mathrm{L}}\alpha \mid \neg\varphi \mid \varphi \wedge \varphi$$
$$\alpha ::= p \mid \neg\alpha \mid \alpha \wedge \alpha \mid \triangle_j\alpha$$

where j ranges over Agt. The language $\mathcal{L}_{1\Box}^{\mathrm{L}}$ consists of formulas that the agent L can check according to their own beliefs. For example, agent L cannot check whether p is true, but they can check (because of introspection) whether $\triangle_{\mathrm{L}}p$ or $\Box_{\mathrm{L}}p$ is true. It follows that they can also check all Boolean combinations over $\triangle_{\mathrm{L}}p$ and $\Box_{\mathrm{L}}p$ (or $\triangle_{\mathrm{L}}\alpha$ and $\Box_{\mathrm{L}}\alpha$ in general). Our shielding constraints will be restricted to the subjective fragment. This is because we need to make sure that each constraint can be checked from the learning agent's perspective.

Example 9. Let $j \neq \mathrm{L}$. Then $\triangle_{\mathrm{L}}p \wedge \Box_{\mathrm{L}}\triangle_j q$ is in $\mathcal{L}_{1\Box}^{\mathrm{L}}$ while p and $\triangle_j q$ are not.

The following proposition implies that only the belief base of agent L is needed to evaluate a subjective formula for L.

Proposition 10. *For all formulas φ in $\mathcal{L}_{1\Box}^{\mathrm{L}}$ and for all σ, σ' with $K_{\mathrm{L}}(\sigma) = K_{\mathrm{L}}(\sigma')$, we have $U, \sigma \models \varphi$ iff $U, \sigma' \models \varphi$.*

Proof. By induction on φ in the grammar of $\mathcal{L}_{1\Box}^{\mathrm{L}}$. □

4.2 Model Checking Problem

In this section we study the problem of checking a formula $\varphi \in \mathcal{L}_{1\Box}$ in a state σ, given a universe $U_\chi = \{\sigma \mid \sigma \models \chi\}$ specified by a formula χ (the *common ground*). We will use the common ground later to inject expert knowledge into the agent.

Definition 11. *Model checking* is the following decision problem:

Input: A formula $\chi \in \mathcal{L}_{\triangle}$, a state $\sigma \in \Sigma$, a formula $\varphi \in \mathcal{L}_{1\Box}$.
Output: Yes if $U_\chi, \sigma \models \varphi$, no otherwise.

Example 12. Let χ be the formula $\neg smell_{0,0} \rightarrow \neg wumpus_{1,0}$. Then we have $U_\chi, \{\triangle\neg smell_{0,0}\} \models \Box\neg wumpus_{1,0}$. Since the agent believes explicitly that there is no smell at $(0,0)$, they can infer that there is no wumpus at position $(1,0)$.

[2] Two calls are independent from each other if the input of either does not depend on the answer of the other.

function $mc(\chi, \sigma, \varphi)$
  match φ **do**
    case p: **return** $p \in \sigma$
    case $\triangle_i\alpha$: **return** $\triangle_i\alpha \in \sigma$
    case $\neg\varphi_1$: **return** not $mc(\chi, \sigma, \varphi_1)$
    case $\varphi_1 \wedge \varphi_2$: **return** $mc(\chi, \sigma, \varphi_1)$ and $mc(\chi, \sigma, \varphi_2)$
    case $\Box_i\alpha$: **return** true iff $(\mathrm{descr}_i(\sigma, Voc) \wedge \chi) \rightarrow \alpha$ is valid in propositional logic, where $Voc = EpAtm(\alpha) \cup EpAtm(\chi)$

Figure 2: Algorithm for checking a formula φ in the state σ given that the universe U_χ is the set of states satisfying χ.

4.3 Algorithm for Model Checking

Figure 2 gives a practical model checking procedure that considers p, $\triangle\alpha$ as atomic propositions. An implicit belief construction $\Box_i\varphi$ is also considered as atomic and is checked by checking that an adequate propositional formula is valid. To define that formula, we introduce, given a state σ and a set $Voc \subseteq Atm$ with $\sigma \subseteq Voc$, the formula $\mathrm{descr}_i(\sigma, Voc)$ defined as follows:

$$\mathrm{descr}_i(\sigma, Voc) := \bigwedge_{\triangle_i\beta\in\sigma} \beta \wedge \bigwedge_{\triangle_i\beta\in\sigma} \triangle_i\beta \wedge \bigwedge_{\triangle_i\beta\in Voc\setminus\sigma} \neg\triangle_i\beta$$

The following proposition shows correctness for the case $\Box_i\psi$ in Algorithm 2.

Proposition 13. *Let $Voc = EpAtm(\alpha) \cup EpAtm(\chi)$ for some $\chi \in \mathcal{L}_{\triangle}$, and $\alpha \in \mathcal{L}_{1\Box}$. Then, the following are equivalent:*

- *$U_\chi, \sigma \models \Box_i\alpha$;*
- *$\big(descr_i(\sigma, Voc) \wedge \chi\big) \rightarrow \alpha$ is propositionally valid.*

Proof. By Definition 6 and the truth condition of $\Box_i\alpha$, $U_\chi, \sigma \models \Box_i\alpha$ means that (1) for all states σ', if $\sigma' \in U_\chi$ and $K_i(\sigma) = K_i(\sigma')$ and $\forall\beta \in K_i(\sigma), \sigma' \models \beta$ then $\sigma' \models \alpha$. Item (1) is equivalent to the fact that (2) for all states σ', if $\sigma' \models \bigwedge_{\triangle_i\beta\in\sigma} \triangle_i\beta \wedge \bigwedge_{\triangle_i\beta\in\sigma} \beta \wedge \chi$ and for all $\triangle_i\beta \notin \sigma$, $\sigma' \models \neg\triangle_i\beta$, then $\sigma' \models \alpha$. Item (2) is equivalent to the fact that (3) $\{\chi\} \cup \bigcup_{\triangle_i\beta\in\sigma} \{\triangle_i\beta\} \cup \bigcup_{\triangle_i\beta\in\sigma} \{\beta\} \cup \bigcup_{\triangle_i\beta\notin\sigma} \{\neg\triangle_i\beta\} \models \alpha$ in propositional logic, that is, the formula on the right side of the symbol $\models$ is a logical consequence of the set of formulas on the left side. Item (3) is equivalent to the fact that (4) $\bigcup_{\triangle_i\beta\in\sigma} \{\triangle_i\beta\} \cup \bigcup_{\triangle_i\beta\in\sigma} \{\beta\} \cup \bigcup_{\triangle_i\beta\notin\sigma} \{\neg\triangle_i\beta\} \models \chi \rightarrow \alpha$ in propositional logic. Item (4) is equivalent to (5) $\bigcup_{\triangle_i\beta\in\sigma} \{\triangle_i\beta\} \cup \bigcup_{\triangle_i\beta\in\sigma} \{\beta\} \cup \bigcup_{\triangle_i\beta\in Voc\setminus\sigma} \{\neg\triangle_i\beta\} \models \chi \rightarrow \alpha$ in propositional logic, due to the fact the following property holds in propositional logic:

$$\bigcup_{p\in X} p \cup \bigcup_{q\notin X'} \neg q \models \omega \text{ iff } \bigcup_{p\in X} p \cup \bigcup_{q\in Atm(\omega)\setminus X'} \neg q \models \omega,$$

for all propositional formulas ω and sets X, X' of atomic propositions. (5) is equivalent to the fact that (6) $\{\chi\} \cup \bigcup_{\triangle_i\beta\in\sigma} \{\triangle_i\beta\} \cup \bigcup_{\triangle_i\beta\in\sigma} \{\beta\} \cup \bigcup_{\triangle_i\beta\in Voc\setminus\sigma} \{\neg\triangle_i\beta\} \models \alpha$. Item (6) is equivalent to the fact that α is a logical consequence of $\big(\mathrm{descr}_i(\sigma, Voc) \wedge \chi\big)$ in propositional logic, hence to the fact that the propositional formula $\big(\mathrm{descr}_i(\sigma, Voc) \wedge \chi\big) \rightarrow \alpha$ is propositionally valid. □

The previous result generalizes the result proved by Lorini [25, Proposition 1] showing that for propositional ω we have "$U_\top, \sigma \models \Box_i\omega$ iff $\omega \in Cn\big(K_i(\sigma)\big)$", where Cn is the classical deductive closure operator over the propositional language.

Proposition 14. *Let $U_\chi = \{\sigma \in \Sigma \mid \sigma \models \chi\}$. Then $mc(\chi, \sigma, \varphi)$ returns true iff $U_\chi, \sigma \models \varphi$.*

Proof. By induction on φ. □

The following proposition characterizes the complexity of the model checking problem for $\mathcal{L}_{1\Box}$.

Proposition 15. *Model checking for $\mathcal{L}_{1\Box}$ is Θ_2^p-complete. Hardness already holds for $\triangle$-free single-agent formulas in $\mathcal{L}^{\mathrm{L}}_{1\Box}$.*

Proof. Membership in Θ_2^p follows from the correctness of Algorithm 2 together with the observation that the calls to an NP-oracle (for the propositional validity problem) are independent from each other. For hardness, it is known that the following decision problem is Θ_2^P-complete [41]:

Input: Propositional formulas $\varphi_1, \ldots, \varphi_n$ such that $\varphi_k \models \varphi_{k+1}$ for all $k \in \{1, \ldots, n-1\}$.
Output: Let k be the smallest index such that φ_k is satisfiable; yes if k is odd, no otherwise.

In this formulation, the problem is a *promise* problem, that is, one in which the valid inputs do not constitute a polytime class. To give a reduction from a more standard (non-promise) problem, we will consider the following:

Input: Propositional formulas $\varphi_1, \ldots, \varphi_n$.
Output: Let k be the smallest index such that $\varphi_1 \vee \cdots \vee \varphi_k$ is satisfiable; yes if k is odd, no otherwise.

It is easy to see that both problems are many-one reducible to each other, and hence the latter is also Θ_2^P-complete.

We now define a many-one reduction from the latter problem to model-checking for the single-agent case with $\triangle$-free formulas in $\mathcal{L}^{\mathrm{L}}_{1\Box}$. Let $\varphi_1, \ldots, \varphi_n$ be arbitrary propositional formulas, and define (χ, σ, φ) by $\chi := \top$, $\sigma := \emptyset$, and $\varphi := \bigvee_{k=1,\ldots,n/2} \big(\Box_{\mathrm{L}} \neg(\varphi_1 \vee \cdots \vee \varphi_{2k}) \wedge \neg\Box_{\mathrm{L}} \neg(\varphi_1 \vee \cdots \vee \varphi_{2k+1})\big)$. The latter formula reads "there is a k such that $\neg(\varphi_1 \vee \cdots \vee \varphi_{2k})$ is necessarily true but $\neg(\varphi_1 \vee \cdots \vee \varphi_{2k+1})$ is possibly false". With this reading in mind, we can see that for $\chi = \top$ and $\sigma = \emptyset$, $U_\chi, \sigma \models \varphi$ holds if and only if there is an odd $i = 2k+1$ such that $\varphi_1 \vee \cdots \vee \varphi_i$ is satisfiable while $\varphi_1 \vee \cdots \vee \varphi_{i-1}$ is not. Since in this case $\varphi_1 \vee \cdots \vee \varphi_j$ is *a fortiori* unsatisfiable for $j < i-1$, we get that $i = 2k+1$ is the first index such that $\varphi_1 \vee \cdots \vee \varphi_i$ is satisfiable, and hence that the reduction is correct. □

Proposition 15 implies that model checking can be performed in practice with a polynomial algorithm that makes parallel calls to a SAT solver.

5 BELIEF BASE POMDPS

Partially Observable Markov Decision Processes (POMDPs) are the model of choice in reinforcement learning when the agent does not have full knowledge of the real world. In this section, after recalling the definition of a POMDP, we explain how to instantiate it in the belief base setting given in Section 3. We end the section with a formal description of the wumpus example in this setting.

5.1 POMDPs

Let us recall the definition of a POMDP. For a set X, we write ΔX for the set of all probability distributions over X.

Definition 16. A *POMDP* is a tuple $(\Sigma, A, t, S, O, r, I, T)$ where Σ is a finite set of states, A is a finite set of *actions*, $t : \Sigma \times A \to \Delta\Sigma$ is a *transition function*, S is a finite set of *observations*, $O : \Sigma \to \Delta S$ is an *observation function*, and $r : \Sigma \times A \times \Sigma \to \mathbb{R}$ is a *reward function*. We assume that it also specifies a distribution of *initial states* $I \in \Delta\Sigma$ and a subset of *terminal states* $T \subseteq \Sigma$.

POMDP states are not directly observed by the agent. In each state σ the agent receives an observation from S, following the probability distribution $O(\sigma)$. The action set A contains abstract action names, such as MoveLeft or Noop. The transition function specifies for each action a and states σ, σ' the probability $t(\sigma, a, \sigma')$ of reaching state σ' from σ via action a. Finally, $r(\sigma, a, \sigma')$ is the reward the learning agent receives for going from state σ to σ' by performing action a.

5.2 BB-POMDPs

In the following, we show how a POMDP $(\Sigma, A, t, S, O, r, I, T)$ is induced from a description based on our logic. We call this POMDP a *Belief Base POMDP* (or BB-POMDP). We assume that the learning agent is a dedicated agent L. Note that our framework allows for additional passive agents whose (true and false) beliefs can be modified by actions and can be relevant to the learning agent's reward. This setting has been discussed in a planning context by Davila et al. [9] as *cognitive planning*. However, we will only consider single-agent examples and leave cognitive planning to future work.

Atoms and states. We assume that a finite set of relevant atoms $RelAtm \subseteq Atm$ is specified by the modeler. Then the set of POMDP states is the set $\Sigma = 2^{RelAtm}$ of all valuations over $RelAtm$. Furthermore, the set of POMDP observations $S = 2^{\{\alpha \mid \triangle_{\mathrm{L}}\alpha \in RelAtm\}}$ is the set of possible belief bases of the learning agent L. Finally, the observation function $O(\sigma)$ assigns probability 1 to the belief base $K_{\mathrm{L}}(\sigma)$ of the learning agent, and 0 to all other belief bases. In the following, we assume all formulas to use only relevant atoms.

Initial and terminal states. The sets of initial and terminal states are specified by formulas $\chi_I, \chi_T \in \mathcal{L}_\triangle$. The set of terminal states is then $T = \{\sigma \in \Sigma \mid \sigma \models \chi_T\}$. The initial state distribution I is the uniform distribution over $\{\sigma \in \Sigma \mid \sigma \models \chi_I\}$. In our implementation, since the $\mathcal{L}_\triangle$ fragment corresponds to propositional logic over Atm, we will use the uniform SAT sampler SPUR [1] to sample initial states.

Common ground. We assume that the *expert knowledge* the agent uses to infer implicit beliefs from their belief base, is given by a *common ground* formula $\chi \in \mathcal{L}_\triangle$. This formula characterizes the context U_χ on which formulas from $\mathcal{L}_{1\Box}$ are evaluated using the model checking algorithm described in Section 4.3.

Actions and transitions. The set A of abstract action names is specified by the modeler. The function $t : \Sigma \times A \to \Sigma$ is then described by a deterministic[3] *action theory*. For each action a and atom x (either a proposition p or an epistemic atom $\triangle_i\alpha$), there is a formula $\varphi^a_x \in \mathcal{L}_{1\Box}$, such that action a will replace the truth value of x by the truth value of formula φ^a_x. Formally, $t(\sigma, a)$ assigns probability 1 to $\sigma' = \{x \in RelAtm \mid U_\chi, \sigma \models \varphi^a_x\}$.

[3] We plan to relax this assumption in future work.

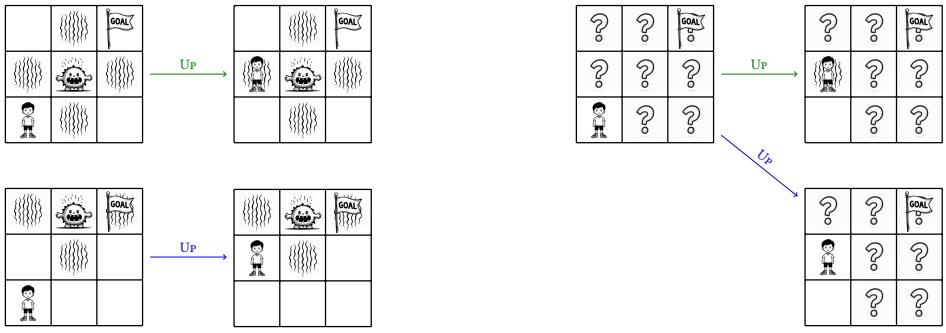

Figure 3: Omniscient observer perspective on the left, agent perspective on the right.

When specifying an action theory, we often omit the definition of some φ^a_x. In these cases we assume that $\varphi^a_x = x$, that is, that a never changes x.

Note that we leave it to the modeler to ensure (if desired) that the action theory cannot lead to inconsistent belief bases.

Example 17. Assume we have $\varphi^a_{\triangle_{\mathrm{L}}p} = \varphi^a_{\triangle_{\mathrm{L}}\neg p} = \top$. In that case, executing a in an arbitrary state will result in a state containing both $\triangle_{\mathrm{L}}p$ and $\triangle_{\mathrm{L}}\neg p$. In this state, agent L has inconsistent beliefs.

Reward. We specify rewards as a set $R \subseteq \mathcal{L}_{1\Box} \times A \times \mathcal{L}_{1\Box} \times \mathbb{R}$. For each tuple $(\varphi, a, \varphi', \rho) \in R$, the idea is that after a transition $\sigma \xrightarrow{a} \sigma'$ in the POMDP, the agent obtains the reward ρ if φ is true in σ and φ' is true in σ'. That is, the reward of the POMDP is defined as

$$r(\sigma, a, \sigma') = \sum_{\substack{(\varphi, a, \varphi', \rho) \in R, \\ \sigma \models \varphi, \sigma' \models \varphi'}} \rho.$$

In POMDPs, the reward is usually assumed to be given to the agent by the environment. An alternative approach are *subjective rewards* [37] which can be computed by the agents given their knowledge. In our framework, we can model both types of reward. The general formulation above can be interpreted as rewards from the environment. If we restrict φ and φ' to the subjective fragment, it can be interpreted as subjective rewards.

5.3 Specifying the Wumpus Example

We now specify the BB-POMDP for Example 1, assuming that there are k wumpuses and the grid is of size $n \times n$. The set *RelAtm* contains for all coordinates $x, y \in \{1, \ldots, n\}$:

- $pos_{x,y}$: The agent is at position (x, y).
- $wumpus_{x,y}$: There is a wumpus at position (x, y).
- $smell_{x,y}$: There is a smell at position (x, y).

It further contains the explicit beliefs $\triangle_{\mathrm{L}} pos_{x,y}$, $\triangle_{\mathrm{L}} wumpus_{x,y}$, $\triangle_{\mathrm{L}} smell_{x,y}$, and $\triangle_{\mathrm{L}} \neg smell_{x,y}$ for all coordinates $x, y \in \{1, \ldots, n\}$.

The set of initial states is defined by a long conjunction χ_I encoding the following constraints: (1) The agent is at position $(1, 1)$, they believe that they are at position $(1, 1)$, and they are at no other position. (2) There are wumpuses on k cells, but none at $(1, 1)$ or (n, n). (3) All cells adjacent to a wumpus are smelly. (4) All cells that are not smelly are not adjacent to a wumpus. (5) The agent explicitly believes $smell_{1,1}$ or $\neg smell_{1,1}$, depending on whether a wumpus is adjacent to $(1, 1)$. (6) The agent has no further beliefs.

The set of terminal states is defined by the formula

$$\chi_T = pos_{n,n} \vee \bigvee_{x,y} (pos_{x,y} \wedge wumpus_{x,y}).$$

The common ground is a formula χ encoding the following expert knowledge: (1) There are wumpuses on k cells. (2) All cells adjacent to a wumpus are smelly. (3) All cells that are not smelly are not adjacent to a wumpus.

The set of actions is $A = \{\textsc{Up}, \textsc{Down}, \textsc{Right}, \textsc{Left}, \textsc{Noop}\}$. We specify the transition function t using a deterministic action theory. As an example we give the action Right:

$$\varphi^{\textsc{Right}}_{pos_{x,y}} = \varphi^{\textsc{Right}}_{\triangle_{\mathrm{L}} pos_{x,y}} = \begin{cases} \bot & x = 1 \\ pos_{x-1,y} & 1 < x < n \\ pos_{n-1,y} \vee pos_{n,y} & x = n \end{cases}$$

$$\varphi^{\textsc{Right}}_{\triangle_{\mathrm{L}} smell_{x,y}} = \begin{cases} \triangle_{\mathrm{L}} smell_{x,y} \vee (pos_{x-1,y} \wedge smell_{x,y}) & 1 < x \leq n \\ \triangle_{\mathrm{L}} smell_{x,y} & \text{otherwise} \end{cases}$$

$$\varphi^{\textsc{Right}}_{\triangle_{\mathrm{L}} \neg smell_{x,y}} = \begin{cases} \triangle_{\mathrm{L}} \neg smell_{x,y} \vee (pos_{x-1,y} \wedge \neg smell_{x,y}) & 1 < x \leq n \\ \triangle_{\mathrm{L}} \neg smell_{x,y} & \text{otherwise} \end{cases}$$

As we can see, movement actions can change the position of the agent, as well as beliefs about its own position and smells. Note the separate cases for the leftmost column (by moving to the right, the agent can never end up in that column) and for the rightmost column (if the agent is already in that column, they will not move further to the right). Previous beliefs about smells are retained, and new beliefs about smells are obtained when the agent moves to a new cell.

We will assume that the agent obtains a reward of 1 if it reaches the goal and a reward of -1 if it runs into the wumpus. That is, the rewards are specified as $R = \{(\top, a, \gamma_{die}, -1) \mid a \in A\} \cup \{(\top, a, \gamma_{goal}, 1) \mid a \in A\}$ with $\gamma_{die} = \bigvee_{x,y} (wumpus_{x,y} \wedge pos_{x,y})$ and $\gamma_{goal} = pos_{n,n}$.

Importantly, while the dynamics in our BB-POMDPs are deterministic (due to the deterministic action theory), they appear nondeterministic to the learning agent. This is illustrated in Figure 3. From the perspective of an omniscient observer, the positions of the wumpuses are known, and it is clear whether there will be a smell at the cell to which the agent moves. However, from the perspective of the agent, the same action can have different outcomes

depending on the unknown position of the wumpus. In the following, we will discuss how to apply reinforcement learning in such a setting.

5.4 Learning in Belief Base POMDPs

In the general POMDP setting, the agent's policy typically depends on the entire past history of actions and observations. For example, there are model-free deep RL approaches that use recurrent neural network architectures to represent policies as a function of past actions and observations [17, 18, 31, 32]. The problem is that observations in general POMDPs must be understood as simple *tokens*, which by themselves (without the history of previous observations and actions taken by the agent) do not contain sufficient information about the current state of the system.

In contrast, we assume that for each state σ of the POMDP, the agent's belief is already fully characterized by their belief base $K(\sigma)$. The objective is thus to compute an optimal policy $\pi : S \to A$ that assigns an action $\pi(s)$ to each belief base $s \in S$.

Since transitions between two belief bases depend on the latent POMDP states, they will be non-Markovian in general. However, in our example, the information the agent has in its belief base about the potential wumpus positions (visited smelly and non-smelly cells) increases monotonically. Since the probabilities with which the wumpuses are initially positioned are well-defined, one could always infer the correct probabilities about the positions for any given belief base, independently of the history of previous states and actions. This implies that the transitions between belief bases, as observed by the learning agent, are Markovian and that we can use model-free learning algorithms for the fully-observable case directly on the belief bases.[4]

What makes learning difficult in our setting is the uncertainty about the latent state. For example, with two wumpuses on a 7x7 grid, there are $\frac{47 \cdot 46}{2} = 1081$ possible configurations of where the two wumpuses can be. This means that, on average, the learning agent experiences each configuration only once every 1081 training episodes. To learn to handle all possible situations, the learning algorithm must either generalize, or the agent would arguably need to explore each configuration several times.

In our experiments, we use a tabular version of Q-learning for simplicity. To be as sample efficient as possible, we use a version of *experience replay* [20] similar to *fitted Q iteration* [15], which always uses the full batch D of previously observed transition-reward tuples in its Q-update step, and never discards any of them. The algorithm[5] is shown in Figure 4 and can be characterized as a growing-batch online reinforcement learning algorithm [19]. The updated Q-function is computed as

$$Q_{new}(s,a) := \mathop{\mathbb{E}}_{(s,a,r,s')\sim D}\left(r + \gamma \cdot \max_{a \in A} Q_{old}(s',a)\right).$$

[4]For the general case where transitions between belief bases are not Markovian, we plan to consider alternative approaches in future work.

[5]We make the simplifying assumption that whether a state σ is terminal can be identified from the state $K(\sigma)$ as observed by the agent. This means that we never get transitions starting from a state s which can also be terminal, and thus we do not need to make a case distinction for terminal states when updating the Q function.

```
function simulate_and_learn(bb_pomdp, num_episodes, len_episode, ϵ)
    initialize D ← ∅ as an empty multiset
    initialize Q(s, a) ← 0 for all s ∈ S, a ∈ A
    for episode ← 1..num_episodes do
        sample initial state σ from I
        s ← K(σ)
        for step ← 1..len_episode do
            if s ∈ T: break
            // perform ϵ-greedy action selection
            if uniform_random(0, 1) < ϵ: sample a from A
            else : sample a from π(s) = arg max_{a∈A} Q(s, a)
            // simulate action and store transition and reward
            sample σ′ from t(σ, a)
            s′, r ← K(σ), r(σ, a, σ′)
            append (s, a, r, s′) to D
            // update state for next simulation step
            σ, s ← σ′, s′
        Update Q(s, a) using all transitions from D
```

Figure 4: The simulation and growing batch Q-learning loop. The simulation is based on the POMDP states, while the agent acts and learns based on their belief states.

An advantage of this approach is that only minimal modifications are required to use neural networks as function approximators (resulting in *Neural Fitted Q iteration* [32]) and subsequently change it to the well-known DQN algorithm [35].

6 CASE STUDY

In addition to the vanilla version of our Wumpus World, we will also define variants that take advantage of the special reasoning capabilities of Belief Base POMDPs, in particular *reward shaping* and *shielding*. We use an exploration constant of $\epsilon = 0.1$ and a discount factor of $\gamma = 0.95$ in all our experiments, with a maximum of 100 steps for each training episode.

6.1 Reward shaping

Explicit and implicit beliefs can be used for *reward shaping*. For example, in the wumpus world, we give the agent a small extra reward of 0.01 whenever it smells a wumpus, i.e. whenever a formula of the form $\triangle smell_{x,y}$ appears for the first time in a state. The rationale is that this encourages the agent to move around (which can be good for finding the way past a wumpus). Since the reward is much smaller than the reward for reaching the goal, it will hopefully not interfere too much with the agent's main objective.

6.2 Shielding

Another useful technique that can leverage explicit and implicit beliefs during learning and execution is *shielding*. It aims to restrict the action that the agent is allowed to take in each step to a *safe* subset: An action a is available exactly in states where some shielding formula $\chi_a \in \mathcal{L}^{\mathrm{L}}_{1\Box}$ is satisfied. Since χ_a is a formula in the subjective fragment, it can be evaluated directly on the belief state of the agent (Proposition 10).

In the wumpus example with shielding, we only want to allow the agent to move to some cell if they implicitly know that there

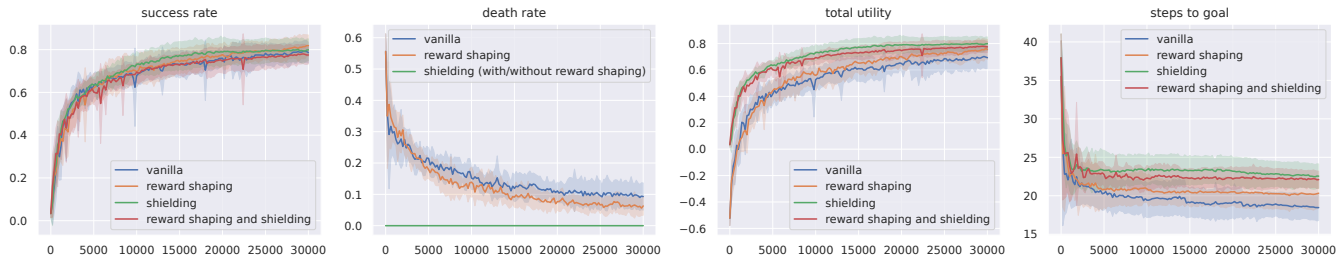

Figure 5: Learning curves for batch Q-learning on the wumpus example (with two wumpuses on a 7×7 grid). Training episodes are on the x axis. The *total utility* metric is the success rate minus the death rate. The shaded areas represent the spread in terms of standard deviation.

is no wumpus on that cell. For the Up action, this is captured by

$$\chi_{\text{Up}} = \bigwedge_{x,y} \left(\triangle pos_{x,y} \rightarrow \Box \neg wumpus_{x,y+1}\right).$$

The shields for the other actions are defined analogously.

6.3 Results

We ran the algorithm given in Figure 4 ten times for each variant to obtain averaged learning curves (Figure 5).[6] For each learning curve, we evaluated every 200th policy on a separately sampled test set of 100 initial states. The learning curves show the performance of the policies in terms of success rate (how often did the agent reach the goal), death rate (how often did the agent run into the wumpus), and total undiscounted utility (the success rate minus the death rate; note that this is equivalent to the reward without reward shaping). Our final graph shows the average number of steps to the goal in episodes where the goal has been reached.

For all variants, the agent learns to reach the goal in a majority of cases. Note that there are wumpus configurations where the goal is not reachable, or where the goal is not reachable without the risk of being eaten by the wumpus, so the theoretical optimum is less than one. As we can see, shielding prevents the agent from ever running into a wumpus, but at the cost of needing more steps to reach the goal. Reward shaping also improves both the success rate and the death rate, but to a lesser extent than shielding, and with a lesser effect on the number of steps needed to reach the goal. Shielding has the best overall utility. A compromise is the combination of reward shaping and shielding, which has a slightly worse overall utility but needs slightly fewer steps to the goal.

7 CONCLUSION AND FUTURE WORK

We have presented an integration of epistemic logic with reinforcement learning, by instantiating POMDPs from a belief base semantics that allows us to represent the learning agent's explicit and implicit beliefs. We have shown that the notion of implicit belief allows an RL agent to be equipped with deductive reasoning capabilities. We implemented our model and experimentally evaluated it on a simple wumpus scenario, demonstrating how it can be successfully used for reward shaping and shielding.

[6]Our implementation can be found at https://github.com/tengesser/epirl.

The paper leaves many possibilities of Belief Base POMDPs unexplored. A big advantage of our approach is that the agents' beliefs are manipulated directly through actions. We might want to leverage this to make the agent learn to actively manage their belief base, for example, by learning when to forget irrelevant explicit beliefs, or when to infer new explicit beliefs from implicit beliefs. We hope that learning to reach the goal while at the same time learning to manage the belief base could lead to better performance and faster convergence. However, for this we arguably need a representation of policies that is capable of generalization. To this end we plan to replace the Q-table in our implementation by a neural network.

Further directions of future work are manifold. First of all, following Lorini [26], we plan to leverage our model in the context of a more realistic dialogue scenario in which an artificial agent has to reason about the human user's beliefs in order to persuade or influence the human through communication. In this context it is crucial for the agent to learn a theory of the interlocutor's mind. For example, the agent may have the goal of persuading the human to use their bike instead of the car for going to work. The agent should learn the quality of an informative action depending on what they believe about the human's beliefs. For instance, they should learn the quality of informing the human that the outside temperature is not too high, when they believe that the human believes that it is not a rainy day.

We also plan to generalize our framework to the multi-agent case in order to model multiple learning agents endowed with deductive reasoning capabilities. This extension would require us to move from standard MDPs and POMDPs to Markov games and from single-agent to multi-agent epistemic logic to come up with an integration of multi-agent epistemic logic with multi-agent RL. This would increase the complexity since model checking for the multi-agent logic of explicit and implicit belief is PSPACE-hard.

ACKNOWLEDGMENTS

Support from the ANR project EpiRL "Epistemic Reinforcement Learning" (grant number ANR-22-CE23-0029) and from the ANR project ALoRS "Action, Logical Reasoning and Spiking networks" (grant number ANR-21-CE23-0018-01) is acknowledged. This research was funded in part by the Austrian Science Fund (FWF) 10.55776/COE12.

REFERENCES

[1] Dimitris Achlioptas, Zayd S. Hammoudeh, and Panos Theodoropoulos. 2018. Fast Sampling of Perfectly Uniform Satisfying Assignments. In *Proc. SAT 2018*. Springer, 135–147.

[2] Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. 2018. Safe Reinforcement Learning via Shielding. In *Proc. AAAI 2018*. AAAI Press, 2669–2678.

[3] Mauricio Araya-López, Olivier Buffet, Vincent Thomas, and François Charpillet. 2010. A POMDP Extension with Belief-dependent Rewards. In *Proc. NIPS 2010*. Curran Associates, Inc., 64–72.

[4] Thomas Bolander and Mikkel Birkegaard Andersen. 2011. Epistemic planning for single and multi-agent systems. *J. Appl. Non Class. Logics* 21, 1 (2011), 9–34.

[5] Thomas Bolander, Tristan Charrier, Sophie Pinchinat, and François Schwarzentruber. 2020. DEL-based epistemic planning: Decidability and complexity. *Artif. Intell.* 287 (2020), 103304.

[6] Thomas Bolander, Martin Holm Jensen, and François Schwarzentruber. 2015. Complexity Results in Epistemic Planning. In *Proc. IJCAI 2015*. AAAI Press, 2791–2797.

[7] Maxime Bouton, Jana Tumova, and Mykel J. Kochenderfer. 2020. Point-Based Methods for Model Checking in Partially Observable Markov Decision Processes. In *Proc. AAAI 2020*. AAAI Press, 10061–10068.

[8] Martin C. Cooper, Andreas Herzig, Faustine Maffre, Frédéric Maris, Elise Perrotin, and Pierre Régnier. 2021. A lightweight epistemic logic and its application to planning. *Artif. Intell.* 298 (2021), 103437.

[9] Jorge Luis Fernandez Davila, Dominique Longin, Emiliano Lorini, and Frédéric Maris. 2021. A Simple Framework for Cognitive Planning. In *Proc. AAAI 2021*. AAAI Press, 6331–6339.

[10] Jorge Luis Fernandez Davila, Dominique Longin, Emiliano Lorini, and Frédéric Maris. 2024. Logic-based cognitive planning for conversational agents. *Auton. Agents Multi Agent Syst.* 38, 1 (2024), 20.

[11] Giuseppe De Giacomo, Luca Iocchi, Marco Favorito, and Fabio Patrizi. 2019. Foundations for Restraining Bolts: Reinforcement Learning with LTLf/LDLf Restraining Specifications. In *Proc. ICAPS 2019*. AAAI Press, 128–136.

[12] Tiago de Lima, Emiliano Lorini, and François Schwarzentruber. 2023. Base-Based Model Checking for Multi-agent only Believing. In *Proc. JELIA 2023*. Springer, 437–445.

[13] Thorsten Engesser, Andreas Herzig, and Elise Perrotin. 2024. Towards Epistemic-Doxastic Planning with Observation and Revision. In *Proc. AAAI 2024*. AAAI Press, 10501–10508.

[14] Thorsten Engesser and Tim Miller. 2020. Implicit Coordination Using FOND Planning. In *Proc. AAAI 2020*. AAAI Press, 7151–7159.

[15] Damien Ernst, Pierre Geurts, and Louis Wehenkel. 2005. Tree-Based Batch Mode Reinforcement Learning. *J. Mach. Learn. Res.* 6 (2005), 503–556.

[16] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. 2003. *Reasoning About Knowledge*. MIT Press, Cambridge, MA, USA.

[17] Matthew J. Hausknecht and Peter Stone. 2015. Deep Recurrent Q-Learning for Partially Observable MDPs. In *Proc. AAAI Fall Symposium 2015*. AAAI Press, 29–37.

[18] Nicolas Heess, Jonathan J. Hunt, Timothy P. Lillicrap, and David Silver. 2015. Memory-based control with recurrent neural networks. *CoRR* abs/1512.04455 (2015), 11 pages.

[19] Sascha Lange, Thomas Gabel, and Martin A. Riedmiller. 2012. Batch Reinforcement Learning. In *Reinforcement Learning*, Marco A. Wiering and Martijn van Otterlo (Eds.). Adaptation, Learning, and Optimization, Vol. 12. Springer, 45–73.

[20] Long Ji Lin. 1992. Self-Improving Reactive Agents Based On Reinforcement Learning, Planning and Teaching. *Mach. Learn.* 8 (1992), 293–321.

[21] Dominique Longin, Emiliano Lorini, and Frédéric Maris. 2020. Beliefs, Time and Space: A Language for the Yōkai Board Game. In *Proc. PRIMA 2020*. Springer, 386–393.

[22] Emiliano Lorini. 2018. In Praise of Belief Bases: Doing Epistemic Logic Without Possible Worlds. In *Proc. AAAI 2018*. AAAI Press, 1915–1922.

[23] Emiliano Lorini. 2019. Exploiting Belief Bases for Building Rich Epistemic Structures. In *Proc. TARK 2019*. 332–353.

[24] Emiliano Lorini. 2020. Rethinking epistemic logic with belief bases. *Artif. Intell.* 282 (2020), 103233.

[25] Emiliano Lorini. 2023. A Rule-Based Modal View of Causal Reasoning. In *Proc. IJCAI 2023*. ijcai.org, 3286–3295.

[26] Emiliano Lorini. 2024. Designing Artificial Reasoners for Communication. In *Proc. AAMAS 2024*. IFAAMAS, 2690–2695.

[27] Emiliano Lorini and Éloan Rapion. 2022. Logical Theories of Collective Attitudes and the Belief Base Perspective. In *Proc. AAMAS 2022*. IFAAMAS, 833–841.

[28] Emiliano Lorini and Fabián Romero. 2019. Decision Procedures for Epistemic Logic Exploiting Belief Bases. In *Proc. AAMAS 2019*. IFAAMAS, 944–952.

[29] Emiliano Lorini and Pengfei Song. 2023. A computationally grounded logic of awareness. *J. Log. Comput.* 33, 6 (2023), 1463–1496.

[30] Benedikt Löwe, Eric Pacuit, and Andreas Witzel. 2011. DEL Planning and Some Tractable Cases. In *Proc. LORI 2011*. Springer, 179–192.

[31] Lingheng Meng, Rob Gorbet, and Dana Kulic. 2021. Memory-based Deep Reinforcement Learning for POMDPs. In *Proc. IROS 2021*. IEEE, 5619–5626.

[32] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nat.* 518, 7540 (2015), 529–533.

[33] Emery A. Neufeld, Agata Ciabattoni, and Radu Florin Tulcan. 2024. Norm Compliance in Reinforcement Learning Agents via Restraining Bolts. In *Proc. JURIX 2024*. IOS Press, 119–130.

[34] Hai Nguyen, Sammie Katt, Yuchen Xiao, and Christopher Amato. 2023. On-Robot Bayesian Reinforcement Learning for POMDPs. In *Proc. IROS 2023*. IEEE, 9480–9487.

[35] Martin A. Riedmiller. 2005. Neural Fitted Q Iteration - First Experiences with a Data Efficient Neural Reinforcement Learning Method. In *Proc. ECML 2005*. Springer, 317–328.

[36] Régis Sabbadin, Jérôme Lang, and Nasolo Ravoanjanahry. 2007. Purely Epistemic Markov Decision Processes. In *Proc. AAAI 2007*. AAAI Press, 1057–1062.

[37] Timothy Schroeder. 2004. *Three Faces of Desire*. Oxford University Press, New York.

[38] Sarath Sreedharan, Tathagata Chakraborti, Christian Muise, and Subbarao Kambhampati. 2024. Planning with mental models - Balancing explanations and explicability. *Artif. Intell.* 335 (2024), 104181.

[39] Robert Stalnaker. 2002. Common Ground. *Linguistics and Philosophy* 25, 5/6 (2002), 701–721.

[40] Leslie G. Valiant. 2003. Three problems in computer science. *J. ACM* 50, 1 (2003), 96–99.

[41] Klaus W. Wagner. 1990. Bounded Query Classes. *SIAM J. Comput.* 19, 5 (1990), 833–846.

[42] Gregory Yob. 1975. Hunt the wumpus. *Creative Computing* 1, 5 (1975), 51–54.