A Logical Modeling of the Yōkai Board Game

Jorge Luis Fernandez Davila, Dominique Longin, Emiliano Lorini, Frédéric Maris

▶ **To cite this version:**

Jorge Luis Fernandez Davila, Dominique Longin, Emiliano Lorini, Frédéric Maris. A Logical Modeling of the Yōkai Board Game. AI Communications, In press, pp.1-32. 10.3233/AIC-230050 . hal-04304902

HAL Id: hal-04304902

https://ut3-toulouseinp.hal.science/hal-04304902v1

Submitted on 24 Nov 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

A Logical Modeling of the Yōkai Board Game

Jorge Fernandez *, Dominique Longin, Emiliano Lorini and Frédéric Maris

[a] *IRIT, Université de Toulouse, CNRS, Toulouse INP, UT3, France*
E-mails: Jorge.Fernandez@irit.fr, Dominique.Longin@irit.fr, Emiliano.Lorini@irit.fr, Frédéric.Maris@irit.fr

Abstract. We present an epistemic language for representing an artificial player's beliefs and actions in the context of the Yōkai board game. Yōkai is a cooperative game which requires a combination of Theory of Mind (ToM), temporal and spatial reasoning to be played effectively by an artificial agent. We show that the language properly accounts for these three dimensions and that its satisfiability problem is NP-complete. This opens up the possibility of exploiting SAT techniques for automating reasoning of an artificial player in the context of the Yōkai board-game.

Keywords: Keyword one, keyword two

1. Introduction

When one wishes to model socio-cognitive agents and, in particular, agents endowed with a Theory of Mind (ToM) who are capable of reasoning about other agents' beliefs, some of the privileged tools are epistemic logic (EL) [1, 2] and its extensions by informative and communicative extensions such as public and private announcements [3–5]. The latter belongs to the Dynamic Epistemic Logic (DEL) family [6].

The major disadvantage of EL and DEL is that they have most of the time a high complexity thereby making them not very well-suited for practical applications. In particular, it has been shown that extending multi-agent EL by simple notions of state eliminating public announcement or arrow eliminating private announcement does not increase its PSPACE complexity (see, e.g., [7, 8]). However, the satisfiability problem of full DEL with public, semi-private and private communicative actions was shown to be NEXPTIME-complete [9]. The situation is even worse in the context of epistemic planning: it is known that epistemic planning in public announcement logic (PAL) is decidable, while it becomes undecidable in full DEL, due to the fact that the epistemic model may grow as a consequence of a private announcement [10].

In [11, 12], a variant of epistemic logic with a semantics exploiting belief bases is introduced. It distinguishes explicit belief from implicit belief. The former is modeled as a fact in an agent's belief base, while the latter is modeled as a fact that is deducible from the agent's explicit beliefs. The main advantages of the belief base semantics for epistemic logic compared to the standard possible world semantics based on multi-relational structures (so-called Kripke models) are (i) its compactness, and (ii) its closeness to the way artificial cognitively-inspired agents are traditionally modeled in AI and in the area of knowledge representation and reasoning (KR) by adopting a database perspective [13]. In [14], it is shown that this variant of epistemic logic provides a valuable abstraction for modeling multi-robot scenarios in which each robot is endowed with a ToM whereby being able to ascribe epistemic states to the other robots and to reason about them.[1]

In this paper, we leverage the belief base semantics for epistemic logic to model interaction in the context of the cooperative board-game Yōkai.[2] We consider its two-player variant in which an artificial agent has to collaborate

*Corresponding author. E-mails: Dominique.Longin@irit.fr, Emiliano.Lorini@irit.fr, Frédéric.Maris@irit.fr.

[1] See also [15, 16] for a DEL-based approach to modeling and implementing ToM on social robots.

[2] https://boardgamegeek.com/boardgame/269146/ykai

with a human agent to win it and to obtain the best score as possible. Yōkai is an interesting testbed for artificial agents, as it covers a lot of epistemic and strategic reasoning aspects as well as planning and belief revision aspects. The idea of testing the performance of artificial agents in the context of cooperative board-games in which ToM reasoning plays a role is not new. Some works exist about modeling and implementing artificial players for the card game Hanabi [17–19]. Yōkai adds to the ToM dimension, which is central in Hanabi, the temporal and spatial dimension. First of all, in Yōkai a player's performance relies on her/its capacity to remember the cards she/it and the other player have seen in the past. Secondly, the players must move cards in a shared space and there are spatial restrictions on card movements that should be taken into account by the players. More generally, the interesting feature of Yōkai, from the point view of KR, is the combination of epistemic, temporal and spatial reasoning that is required to completely apprehend all the game facets and dimensions.

The main novelty of our approach to modeling artificial board-game players is the use of SAT-based techniques. Specifically, the language we present for representing the artificial player's beliefs about the static and dynamic aspects of the game as well as about the human player's beliefs has the same complexity as SAT and can be polynomially translated into a propositional logic language. This opens up the possibility of exploiting SAT techniques for automating reasoning of the artificial player in the context of the Yōkai board-game.

The paper is organized as follows. In Section 2, we explain the rules of Yōkai and clarify the representation and reasoning requirements that are necessary for the artificial player to be able to play the game in a clever way. In Section 3, we introduce the specification language for modeling the artificial player's actions and beliefs about the game properties and about the human player's beliefs. It is a timed language for explicit and implicit belief with a semantics exploiting belief bases. The main novelty compared to the epistemic language presented in [11] is the temporal component: the artificial player modeled in the language has knowledge about the current time of the game and beliefs about current and past beliefs of the human player. In Section 4, we show how the formal language presented in the previous section can be used to endow the artificial agent $\mathfrak{m}$ with the capacity (i) to select an executable action for achieving a certain goal, and (ii) to revise its beliefs during its interaction with agent $\mathfrak{h}$. Section 5 is devoted to the encoding of the game in the language. We first model the static aspects, namely, the artificial agent's beliefs about the rules and properties of the game. Then, we focus on the dynamic aspects by modeling the artificial player's actions and how they are used for planning a sequence of moves at a given round of the game. Finally, Section 6 briefly describes the implementation of our logic within an agent architecture, as well as some experiments that have been conducted in different configurations.

2. Game description

In this section, we explain the rule of the Yōkai game. As pointed out in the introduction, we only consider the two-player variant of the game with an artificial and a human player.

Colored cards and actions. There are 4 types of card, red, yellow, green and blue cards, and 4 cards for each color. This gives us a total amount of 16 cards. The cards are placed face down in a grid of size 4×4 in a random way, so that the players cannot see their colors. The goal of the game is to gather together the cards of the same color. Figure 1a illustrates a winning configuration of the game.

(a) Winning configuration (b) Illegal move

Figure 1. Examples of game configuration

The players play sequentially in each round of the game. At each round, each player has to play the following 4 actions in the order she/it prefers:

(1) to look at two cards privately (2 actions),
(2) to move one card from its current position to a new position adjacent to another card (that is, linked to the latter by one of its sides) and without separating the cards into two disjointed groups,
(3) either to activate a hint from the set of available hints (see next paragraph) or to disclose an information by marking one card with an active hint.

Figure 1b portrays an example of illegal move. Indeed, the move will make the first and second cards on the left bottom corner of the grid separated from the rest of the cards.

Hints. Three types of hint are available: 1-color, 2-color and 3-color hints. Specifically, 1-color, 2-color and 3-color hints indicate, respectively, a single color, two colors and three colors in the set {red, green, blue, yellow}. There are 14 different hints available, but only seven hints are randomly selected at the beginning of the game: two occurrences of 1-color hint, three occurrences of 2-color hint and two occurrences of 3-color hint.

A hint can be used after being activated, and only available hints can be activated. Moreover, it can be used only once throughout the game. The state of a hint is unique: it is either available, activated, or marking. In particular, after a hint has been activated and used to mark a card, it is no longer available or activated. For example, suppose a player just looked at one card and discovers it is a red card. She can mark it with an active 2-color hint of type red/green to inform the other player that it is either a red card or a green card.

A hint is properly used if one of its colors matches with the color of the card it marks. For example, if a player uses an active 2-color card of type green/blue to mark a blue card, then the hint is properly used. Conversely, it the player uses it to mark a red card, then the hint is improperly used. Note that unless a mistake has been made, hints must be used properly.

Finally, when an active hint marks a card, that card can no longer be moved or observed.

End of the game. When a player thinks that the goal state is achieved, she/it can decide to stop the game. Otherwise, the game ends when all hints have been used by the players. At the end of the game, the cards are turned face up. If the goal state is achieved, the players win the game. Otherwise, they lose it. In case of win, the final score is calculated as follows: $score = x_1 - x_2 + 2x_3 + 5x_4$ where x_1 is the number of properly used hints, x_2 is the number of improperly used hints, x_3 is the number of activated hints that were not used by the players, and x_4 is the number of non-activated hints. The final score ranges in the interval $[-7, 35]$.

Requirements. As emphasized in the introduction, in order to be able to reason about the static and dynamic aspects of the game, a player must have beliefs about:

- the other player's actual beliefs (*ToM reasoning*);
- the current positions of the cards and the executable card movements, given the current spatial configuration of the game (*spatial reasoning*);
- the color of the cards she/it observed in the past as well as the other player's past observations (*temporal reasoning*).

3. A timed language for explicit and implicit belief

This section presents a two-agent timed variant of the language and the semantics of the logic of explicit and implicit belief presented in [11]. The two agents are the artificial agent (or machine) $\mathfrak{m}$ and the human user $\mathfrak{h}$. Agents $\mathfrak{m}$ and $\mathfrak{h}$ are treated asymmetrically. Our language allows us to represent (i) $\mathfrak{h}$'s explicit beliefs at different points in a game sequence, and (ii) $\mathfrak{m}$'s actual explicit and implicit beliefs, namely, $\mathfrak{m}$'s explicit and implicit beliefs at the current time point of the game sequence. Following [11], explicit beliefs are defined to be beliefs in an agent's belief base, while implicit beliefs are those beliefs that are derivable from the agent's explicit beliefs.

We first present the static language in which agent $\mathfrak{m}$'s beliefs do not change. Then, we present a dynamic extension in which agent $\mathfrak{m}$'s belief base can be expanded by new information.

3.1. Static language

Assume a countably infinite set of atomic propositions *ATM*. We define the language in two steps.

We first define the language $\mathcal{L}_0(ATM)$ by the following grammar in BNF:

$$\alpha \quad ::= \quad p^t \mid \triangle^t_{\mathfrak{h}}\,\alpha \mid now^{\geqslant t} \mid \neg\alpha \mid \alpha_1 \wedge \alpha_2 \mid \triangle_{\mathfrak{m}}\,\alpha$$

where p ranges over *ATM* and t ranges over $\mathbb{N}$. $\mathcal{L}_0(ATM)$ is the language for representing agent $\mathfrak{h}$'s timed explicit beliefs and agent $\mathfrak{m}$'s actual explicit beliefs. Specifically, the formula $\triangle^t_{\mathfrak{h}}\,\alpha$ is read "agent $\mathfrak{h}$ explicitly believes that α at time t", whilst $\triangle_{\mathfrak{m}}$ is read "agent $\mathfrak{m}$ actually has the explicit belief that α". The beliefs of the machine $\mathfrak{m}$ are not indexed by time because our language only allows us to talk about the beliefs of $\mathfrak{m}$ in the present tense. (It is not a question here of a technical impossibility, but the language thus extended would introduce a greater expressiveness of the language not necessary here to model our problem.) Furthermore, it is important to note that past beliefs are accessible from previous belief states (see below the section on revision of the belief base). Atomic propositions are assumed to be timed: p^t is read "atomic proposition p is true at time t". Finally, formula $now^{\geqslant t}$ provides information about the current time point. It is read "the actual time of the game play is at least t".

Then, we define $\mathcal{L}^T_0(ATM)$ to be the subset $\mathcal{L}_0(ATM)$ including only timed formulas, that is:

$$\mathcal{L}^T_0(ATM) = \{p^t : p \in ATM \text{ and } t \in \mathbb{N}\} \cup \{\triangle^t_{\mathfrak{h}}\,\alpha : \alpha \in \mathcal{L}_0(ATM) \text{ and } t \in \mathbb{N}\} \cup \{now^{\geqslant t} : t \in \mathbb{N}\}.$$

Elements of $\mathcal{L}^T_0(ATM)$ are noted $x, y, \ldots$

The language $\mathcal{L}(ATM)$ extends the language $\mathcal{L}_0(ATM)$ by a modal operator of implicit belief for agent $\mathfrak{m}$ and is defined by the following grammar:

$$\varphi \quad ::= \quad \alpha \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \Box_{\mathfrak{m}}\,\alpha,$$

where α ranges over $\mathcal{L}_0(ATM)$. For notational convenience we write $\mathcal{L}_0$ instead of $\mathcal{L}_0(ATM)$, $\mathcal{L}^T_0$ instead of $\mathcal{L}^T_0(ATM)$ and $\mathcal{L}$ instead of $\mathcal{L}(ATM)$, when the context is unambiguous. The formula $\Box_{\mathfrak{m}}\,\alpha$ is read "agent $\mathfrak{m}$ actually has the implicit belief that α". The other Boolean constructions $\top$, $\bot$, $\vee$, $\rightarrow$ and $\leftrightarrow$ are defined in the standard way. Notice that only formulas from the sublanguage $\mathcal{L}_0$ can be in the scope of the implicit belief operator $\Box_{\mathfrak{m}}$. Therefore, nesting of this operator is not allowed (e.g., $\Box_{\mathfrak{m}} \neg \Box_{\mathfrak{m}}\, p^t$ is not a well-formed formula). As we will show at the end of the section, this syntactic restriction on our language is useful to make the complexity of its satisfiability problem the same as the complexity of SAT.

The interpretation of the language $\mathcal{L}$ exploits the notion of belief base. While the notions of possible world and epistemic alternative are primitive in the standard Kripke semantics for epistemic logic [1], they are defined from the primitive concept of belief base in our semantics.

Definition 1 (State). *A state is a tuple $S = (B, V)$ where (i) $B \subseteq \mathcal{L}_0$ is agent $\mathfrak{m}$'s belief base (or, agent $\mathfrak{m}$'s subjective view of the actual situation), (ii) $V \subseteq \mathcal{L}^T_0$ is the actual situation, and such that, for every $t, t' \in \mathbb{N}$,*

$$now^{\geqslant 0} \in V, \tag{1}$$

$$\text{if } now^{\geqslant t} \in V \text{ and } t' \leqslant t \text{ then } now^{\geqslant t'} \in V, \tag{2}$$

$$now^{\geqslant t} \in V \text{ iff } now^{\geqslant t} \in B. \tag{3}$$

The set of all states is noted **S**.

Conditions (1) and (2) in the previous definition guarantees time consistency, namely, that the current time should be at least 0 and that if the current time is at least t and $t' \leqslant t$, then it should be at least t'. Conditions (3) captures agent $\mathfrak{m}$'s time-knowledge, namely, the assumption that $\mathfrak{m}$ has complete information about the current time.

The sublanguage $\mathcal{L}_0(ATM)$ is interpreted w.r.t. states as follows.

Definition 2 (Satisfaction)**.** *Let* $S = (B, V) \in \mathbf{S}$*. Then:*

$$S \models x \iff x \in V,$$
$$S \models \neg\alpha \iff S \not\models \alpha,$$
$$S \models \alpha_1 \wedge \alpha_2 \iff S \models \alpha_1 \text{ and } S \models \alpha_2,$$
$$S \models \triangle_{\mathfrak{m}} \alpha \iff \alpha \in B.$$

Observe in particular the set-theoretic interpretation of the explicit belief operator for agent $\mathfrak{m}$: agent $\mathfrak{m}$ actually has the explicit belief that α if and only if α is included in her actual belief base. This highlights the asymmetry between agent $\mathfrak{m}$ and agent $\mathfrak{h}$ in our semantics. We adopt agent $\mathfrak{m}$'s *internal* perspective, that is, the point of view of its belief base.[3] On the contrary, agent $\mathfrak{h}$'s explicit beliefs are modeled from an *external* point of view and semantically interpreted in the same way as the other timed formulas in $\mathcal{L}_0^T(ATM)$.

A multi-agent belief model (MAB) is defined to be a state supplemented with a set of states, called *context*. The latter includes all states that are compatible with agent $\mathfrak{m}$'s background knowledge.

Definition 3 (Model)**.** *A model is a pair* (S, Cxt)*, where* $S \in \mathbf{S}$ *and* $Cxt \subseteq \mathbf{S}$*. The class of all models is noted* $\mathbf{M}$*.*

Note that we do not impose that $S \in Cxt$. When $Cxt = \mathbf{S}$ then (S, Cxt) is said to be *complete*, since $\mathbf{S}$ is conceivable as the complete (or universal) context which contains all possible states.

Definition 4 (Epistemic alternatives)**.** *We define* $\mathcal{R}$ *to be the binary relation on the set* $\mathbf{S}$ *such that, for all* $S = (B, V), S' = (B', V') \in \mathbf{S}$*:*

$$S\mathcal{R}S' \quad \text{if and only if} \quad \forall\alpha \in B : S' \models \alpha.$$

$S\mathcal{R}S'$ means that S' is an epistemic alternative for the artificial agent $\mathfrak{m}$ at S. So $\mathfrak{m}$'s set of epistemic alternatives at S, noted $\mathcal{R}(S) = \{S' \in \mathbf{S} : S\mathcal{R}S'\}$, includes exactly those states that satisfy $\mathfrak{m}$'s explicit beliefs.

Definition 5 extends Definition 2 to the full language $\mathcal{L}$. Its formulas are interpreted with respect to models. We omit Boolean cases that are defined in the usual way.

Definition 5 (Satisfaction, cont.)**.** *Let* $(S, Cxt) \in \mathbf{M}$*. Then:*

$$(S, Cxt) \models \alpha \iff S \models \alpha;$$
$$(S, Cxt) \models \Box_{\mathfrak{m}} \varphi \iff \forall S' \in Cxt, \text{ if } S\mathcal{R}S' \text{ then } (S', Cxt) \models \varphi.$$

A formula $\varphi \in \mathcal{L}$ is valid in the class $\mathbf{M}$, noted $\models_{\mathbf{M}} \varphi$, if and only if $(S, Cxt) \models \varphi$ for every $(S, Cxt) \in \mathbf{M}$; it is satisfiable in $\mathbf{M}$ if and only if $\neg\varphi$ is not valid in $\mathbf{M}$. As the following theorem indicates, the satisfiability problem for $\mathcal{L}(ATM)$ has the same complexity as SAT. The sketch of proof of the theorem is given in the Appendix B at the end of the paper.

Theorem 1. *Checking satisfiability of* $\mathcal{L}(ATM)$ *formulas in the class* $\mathbf{M}$ *is an NP-complete problem.*

[3] See [20] for an in-depth logical analysis of the internal perspective on modeling knowledge and belief.

3.2. Dynamic extension

Let us now move from a static to a dynamic perspective by presenting a extension of the language $\mathcal{L}(ATM)$ with belief expansion operators. Specifically, we introduce the following language $\mathcal{L}^+(ATM)$, or simply $\mathcal{L}^+$:

$$\varphi \quad ::= \quad \alpha \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \Box_{\mathfrak{m}}\, \alpha \mid [+^{t}_{\mathfrak{m}}\alpha]\varphi,$$

where α ranges over $\mathcal{L}_0$ and t ranges over $\mathbb{N}$. The formula $[+^{t}_{\mathfrak{m}}\alpha]\varphi$ is read "φ holds after agent $\mathfrak{m}$ has privately learned that α and that the current time is at least t" or simply "φ holds after agent $\mathfrak{m}$ has privately learned that α at time at least t".

Our extension has the following semantics relative to a model:

Definition 6 (Satisfaction relation, cont.)**.** *Let $S = (B, V) \in \mathbf{S}$ and $(S, Cxt) \in \mathbf{M}$. Then,*

$$(S, Cxt) \models [+^{t}_{\mathfrak{m}}\alpha]\varphi \Longleftrightarrow (S^{+^{t}_{\mathfrak{m}}\alpha}, Cxt) \models \varphi$$

with

$$S^{+^{t}_{\mathfrak{m}}\alpha} = (B^{+^{t}_{\mathfrak{m}}\alpha}, V^{+^{t}_{\mathfrak{m}}\alpha}),$$
$$V^{+^{t}_{\mathfrak{m}}\alpha} = V \cup \{now^{\geqslant t'} : t' \leqslant t\},$$
$$B^{+^{t}_{\mathfrak{m}}\alpha} = B \cup \{\alpha\} \cup \{now^{\geqslant t'} : t' \leqslant t\}.$$

Intuitively speaking, agent $\mathfrak{m}$'s private learning that α at time at least t simply consists in (i) adding the information α to $\mathfrak{m}$'s belief base, and (ii) moving the objective time and $\mathfrak{m}$'s subjective view of time to index t.

As the following proposition indicates, the dynamic semantics given in Definition 6 is well-defined, as it guarantees that the structure resulting from a belief expansion operation belongs to the class $\mathbf{M}$, if the initial structure also belongs to $\mathbf{M}$.

Proposition 1. *Let $(S, Cxt) \in \mathbf{M}$. Then, $(S^{+^{t}_{\mathfrak{m}}\alpha}, Cxt) \in \mathbf{M}$.*

Satisfiability and validity of formulas in $\mathcal{L}^+$ relative to the class $\mathbf{M}$ are analogous to satisfiability and validity for formulas in $\mathcal{L}(ATM)$ defined above. Interestingly, adding belief expansion operators to the language $\mathcal{L}$ does not increase the complexity of the corresponding satisfiability problem. The sketch of proof of the theorem is given in the Appendix C at the end of the paper.

Theorem 2. *Checking satisfiability of $\mathcal{L}^+(ATM)$ formulas in the class $\mathbf{M}$ is an NP-complete problem.*

It is useful to define the concept of logical consequence for the language $\mathcal{L}^+(ATM)$ which will be used at a later stage to define the action selection problem for the artificial agent $\mathfrak{m}$. Let Σ be a finite subset of $\mathcal{L}_0(ATM)$ and let $\varphi \in \mathcal{L}^+(ATM)$. We say that φ is a logical consequence of Σ in the class $\mathbf{M}$, noted $\Sigma \models_{\mathbf{M}} \varphi$, if and only if, for every $(B, Cxt) \in \mathbf{M}$ such that $Cxt \subseteq \mathbf{S}(\Sigma)$ we have $(B, Cxt) \models \varphi$, with $\mathbf{S}(\Sigma) = \{B \in \mathbf{S} : \forall \alpha \in \Sigma, B \models \alpha\}$. We say that φ is Σ-satisfiable in the class $\mathbf{M}$ if and only if, $\neg\varphi$ is not a logical consequence of Σ in $\mathbf{M}$. Clearly, φ is valid if and only if φ is a logical consequence of $\emptyset$, and φ is satisfiable if and only if φ is $\emptyset$-satisfiable.

As the following deduction theorem indicates, the logical consequence problem with a finite set of premises can be reduced to the satisfiability problem.

Theorem 3. *Let $\varphi \in \mathcal{L}^+(ATM)$ and let $\Sigma \subset \mathcal{L}_0(ATM)$ be finite. Then, $\Sigma \models_{\mathbf{M}} \varphi$ if and only if $\models_{\mathbf{M}} \bigwedge_{\alpha\in\Sigma} \Box_{\mathfrak{m}}\, \alpha \rightarrow \varphi$.*

4. Artificial agent architecture

In this section, we are going to show how the formal language presented in Section 3 can be used to endow the artificial agent $\mathfrak{m}$ with the capacity (i) to select an executable action for achieving a certain goal, and (ii) to revise its beliefs during its interaction with agent $\mathfrak{h}$.

4.1. Action selection

Let $Act_{\mathfrak{m}} = \{+^t_{\mathfrak{m}}\alpha : \alpha \in \mathcal{L}_0, t \in \mathbb{N}\}$ be the set of belief expansion events, or informative actions, formally defined in Section 3.2. Such informative actions have executability preconditions that are specified by the following function $\mathcal{P} : Act_{\mathfrak{m}} \longrightarrow \mathcal{L}(ATM)$. We define the following operator of successful occurrence of an event in $Act_{\mathfrak{m}}$:

$$\langle\langle +^t_{\mathfrak{m}}\alpha\rangle\rangle\varphi \quad \stackrel{\text{def}}{=} \quad \mathcal{P}(+^t_{\mathfrak{m}}\alpha) \wedge [+^t_{\mathfrak{m}}\alpha]\varphi.$$

The formula $\langle\langle +^t_{\mathfrak{m}}\alpha\rangle\rangle\varphi$ in $\mathcal{L}(ATM)$ has to be read "agent $\mathfrak{m}$ can privately learn that α at time t and φ holds after the occurrence of this learning event".

Suppose the artificial agent $\mathfrak{m}$ has a goal represented by a formula in the language $\mathcal{L}_0(ATM)$. An action selection problem for $\mathfrak{m}$ just consists in finding an executable action in its finite action repertoire whose execution guarantees that the goal will be achieved.

Definition 7 (Action selection problem)**.** *An action selection problem is a tuple $\langle \Sigma, Op, \alpha_G \rangle$ where:*

- $\Sigma \subset \mathcal{L}_0(ATM)$ *is a finite set of agent* $\mathfrak{m}$*'s available information,*
- $Op \subset Act_{\mathfrak{m}}$ *is a finite set of operators representing agent* $\mathfrak{m}$*'s action repertoire,*
- $\alpha_G \in \mathcal{L}_0(ATM)$ *is agent* $\mathfrak{m}$*'s goal.*

A solution to the action selection problem $\langle \Sigma, Op, \alpha_G \rangle$ is an action ϵ in Op such that $\Sigma \models_{\mathbf{M}} \langle\langle \epsilon \rangle\rangle \Box_{\mathfrak{m}} \alpha_G$. In other words, a solution to the action selection problem is an executable action in agent $\mathfrak{m}$'s repertoire such that if agent $\mathfrak{m}$ has the information in Σ at its disposal then, after executing the action, it will believe that its goal α_G is achieved.

An action selection problem can be seen as a limit case of a planning problem with a single action to be selected from the action repertoire, instead of a sequence of actions as in the general planning domain. Thanks to Theorems 2 and 3, we can conclude that the action selection problem is NP-complete too.

4.2. Belief change

A crucial component of the action selection problem defined in Definition 7 is agent $\mathfrak{m}$'s available information Σ. The latter includes the information about the rules of the game as well as information about the actual configuration of the game and agent $\mathfrak{h}$'s beliefs. Some of this information evolve during the game. In this section we describe agent $\mathfrak{m}$'s belief change mechanisms that are responsible for inducing this kind of information change.

We distinguish belief update from belief revision. Belief update consists in making the agent $\mathfrak{m}$'s explicit beliefs evolve from a time t to the next time $t+1$. Belief revision is the result of agent $\mathfrak{m}$ learning a new fact and adding a new piece of information to its belief base. We assume agent $\mathfrak{m}$'s available information Σ used in the action selection phase is split into two sets $\Sigma_c, \Sigma_m \subseteq \mathcal{L}_0$. They denote, respectively, the core (or, immutable) information in agent $\mathfrak{m}$'s belief base and the volatile (or, mutable) information in agent $\mathfrak{m}$'s belief base. Agent $\mathfrak{m}$'s core beliefs are stable and do not change under belief revision. On the contrary, volatile beliefs can change due to a belief revision operation. Moreover, we assume the mutable belief base Σ_m is the union of two sets Σ_h and Σ_f. Σ_h includes all hypothetical information that agent $\mathfrak{m}$ can use for reasoning, while Σ_f contains agent $\mathfrak{m}$'s factual information, that is, all facts that agent $\mathfrak{m}$ observes during its interaction with agent $\mathfrak{h}$ through the game. We assume information in Σ_f has priority over information in Σ_h, in the sense that in case its belief base becomes inconsistent, agent $\mathfrak{m}$ prefers to remove information from Σ_h than to remove information from Σ_f in order to restore consistency.

We assume agent $\mathfrak{m}$'s mutable belief base is *present-focused*, in the sense that it contains information about the present time point t and no other time point t' in the future or in the past of t. Let us define this notion formally. The following function specifies the set of time indexes appearing in a $\mathcal{L}_0$-formula:

$$\begin{aligned} time(p^t) &= \{t\}, \\ time(\triangle^t_{\mathfrak{h}}\,\alpha) &= \{t\} \cup time(\alpha), \\ time(now^{\geqslant t}) &= \{t\}, \\ time(\neg\alpha) &= time(\alpha), \\ time(\alpha_1 \wedge \alpha_2) &= time(\alpha_1) \cup time(\alpha_2). \end{aligned}$$

The fact that Σ_m is *present-focused* means that there exists $\Sigma \subseteq \mathcal{L}_0$ and $t \in \mathbb{N}$ such that:

- $\forall \alpha \in \Sigma, time(\alpha) = \{t\}$,
- $\Sigma_m = \Sigma \cup \{now^{\geqslant t'} : t' \leqslant t\}$.

4.2.1. Belief update

We see belief update as a function *Upd* which takes a triple $(\Sigma_c, \Sigma_h, \Sigma_f)$ specifying the core belief base, the hypothetical mutable belief base and the factual mutable belief base of agent $\mathfrak{m}$ as input and returns a new triple $(\Sigma'_c, \Sigma'_h, \Sigma'_f)$ as output.

We consider a specific update function which simply increments of one unit the time indexes of all formulas appearing in agent $\mathfrak{m}$'s mutable belief base, while keeping agent $\mathfrak{m}$'s core belief base unchanged. That is, let *incrt* be the function devoted to increment the time indexes of a $\mathcal{L}_0$-formula of one unit:

$$\begin{aligned} incrt(p^t) &= p^{t+1}, \\ incrt(\triangle^t_{\mathfrak{h}}\,\alpha) &= \triangle^{t+1}_{\mathfrak{h}} incrt(\alpha), \\ incrt(\triangle_{\mathfrak{m}}\,\alpha) &= \triangle_{\mathfrak{m}} incrt(\alpha), \\ incrt(now^{\geqslant t}) &= now^{\geqslant t+1}, \\ incrt(\neg\alpha) &= \neg incrt(\alpha), \\ incrt(\alpha_1 \wedge \alpha_2) &= incrt(\alpha_1) \wedge incrt(\alpha_2). \end{aligned}$$

For each finite $X \subseteq \mathcal{L}_0$, we define $incrt(X) = \{incrt(\alpha) : \alpha \in X\}$.

We stipulate that $Upd(\Sigma_c, \Sigma_h, \Sigma_f) = (\Sigma'_c, \Sigma'_h, \Sigma'_f)$ if and only if:

- $\Sigma'_c = \Sigma_c$,
- $\Sigma'_h = incrt(\Sigma_h)$,
- $\Sigma'_f = incrt(\Sigma_f) \cup \{now^{\geqslant 0}\}$.

It is straightforward to verify that $\Sigma'_m = \Sigma'_h \cup \Sigma'_f$ is present-focused since $\Sigma_m = \Sigma_h \cup \Sigma_f$ is present-focused too.

The belief update function so defined relies on two assumptions that make perfect sense in the context of the interaction between agent $\mathfrak{m}$ and agent $\mathfrak{h}$ in the Yōkai game. First, agent $\mathfrak{m}$ does not keep in its memory information about past facts. We make this assumption since we want to maintain agent $\mathfrak{m}$'s mutable belief base manageable and to avoid that it constantly increases through the game. Secondly, agent $\mathfrak{m}$ assumes that *by default* (i) the world does not change and agents $\mathfrak{h}$ and $\mathfrak{m}$ do not forget what they believe, and (ii) agents $\mathfrak{h}$ and $\mathfrak{m}$ have common knowledge that (i).[4] Nonetheless, after having updated its belief base, agent $\mathfrak{m}$ can learn new information which is incompatible with its actual beliefs. In this case, it will need to revise its beliefs. How agent $\mathfrak{m}$ must revise its beliefs is the content of the next section.

[4] 'By default' means "if no agent acts".

4.2.2. Belief revision

At each time step of the game agent $\mathfrak{m}$ performs belief revision after belief update. We assume that in the belief revision phase formulas in the language $\mathcal{L}_0$ are treated as atomic formulas. In particular, let $\mathcal{L}_{\mathsf{PROP}}$ be the propositional language built from the following set of atomic propositions:

$$PROP = \{\tau_x : x \in \mathcal{L}_0^T(ATM)\} \cup \{\tau_{\triangle_{\mathfrak{m}}\,\alpha} : \triangle_{\mathfrak{m}}\,\alpha \in \mathcal{L}_0(ATM)\}.$$

The following translation transforms each formula in $\mathcal{L}_0$ into its propositional logic counterpart in the language $\mathcal{L}_{\mathsf{PROP}}$:

$$\begin{aligned} tr_{\mathsf{PROP}}(x) =& \tau_x \text{ for } x \in \mathcal{L}_0^T(ATM),\\ tr_{\mathsf{PROP}}(\triangle_{\mathfrak{m}}\,\alpha) =& \tau_{\triangle_{\mathfrak{m}}\,\alpha},\\ tr_{\mathsf{PROP}}(\neg\alpha) =& \neg tr_{\mathsf{PROP}}(\alpha),\\ tr_{\mathsf{PROP}}(\alpha_1 \wedge \alpha_2) =& tr_{\mathsf{PROP}}(\alpha_1) \wedge tr_{\mathsf{PROP}}(\alpha_2). \end{aligned}$$

For each finite $X \subseteq \mathcal{L}_0$, we define $tr_{\mathsf{PROP}}(X) = \{tr_{\mathsf{PROP}}(\alpha) : \alpha \in X\}$.

We say that X is propositionally consistent if and only if $\bot \notin Cn\big(tr_{\mathsf{PROP}}(X)\big)$, where Cn is the classical deductive closure operator over the propositional language $\mathcal{L}_{\mathsf{PROP}}$. Clearly, the latter is equivalent to saying that $\bigwedge_{\alpha\in X} tr_{\mathsf{PROP}}(\alpha)$ is satisfiable in propositional logic.

Let $\Sigma_{input} \subseteq \mathcal{L}_0$ be agent $\mathfrak{m}$'s input information set. We see belief revision as a function *Rev* which takes a quadruple $(\Sigma_c, \Sigma_h, \Sigma_f, \Sigma_{input})$ specifying the core belief base, the hypothetical mutable belief base and the factual mutable belief base of agent $\mathfrak{m}$ together with the input information set and returns a triple $(\Sigma_c', \Sigma_h', \Sigma_f')$.

The revision of $(\Sigma_c, \Sigma_h, \Sigma_f)$ by input Σ_{input}, noted $Rev(\Sigma_c, \Sigma_h, \Sigma_f, \Sigma_{input})$, is formally defined as follows:

(1) if $\Sigma_c \cup \Sigma_{input}$ is not propositionally consistent then $Rev(\Sigma_c, \Sigma_h, \Sigma_f, \Sigma_{input}) = (\Sigma_c, \Sigma_h, \Sigma_f)$,
(2) otherwise, $Rev(\Sigma_c, \Sigma_h, \Sigma_f, \Sigma_{input}) = (\Sigma_c', \Sigma_h', \Sigma_f')$, with

$$\begin{aligned} \Sigma_c' &= \Sigma_c,\\ \Sigma_f' &= \bigcap_{X\in MCS(\Sigma_c,\Sigma_f,\Sigma_{input})} X,\\ \Sigma_h' &= \bigcap_{X\in MCS(\Sigma_c\cup\Sigma_f',\Sigma_h,\emptyset)} X, \end{aligned}$$

where for all $\Sigma, \Sigma', \Sigma'' \subseteq \mathcal{L}_0$, we have $X \in MCS(\Sigma, \Sigma', \Sigma'')$ if and only if:

- $X \subseteq \Sigma' \cup \Sigma''$,
- $\Sigma'' \subseteq X$,
- $X \cup \Sigma$ is propositionally consistent, and
- there is no $X' \subseteq \Sigma' \cup \Sigma''$ such that $X \subset X'$ and $X' \cup \Sigma$ is propositionally consistent.

The revision function *Rev* has the following effects on agent $\mathfrak{m}$'s beliefs. First of all, the core belief base is not modified.

Secondly, the input Σ_{input} is added to the factual mutable belief base only if it is consistent with the core beliefs. In the latter case, the updated factual mutable belief base is equal to the intersection of the subsets of the factual mutable belief base which are maximally consistent with respect to the core belief base and which include the input Σ_{input}. This guarantees that belief revision satisfies minimal change for factual information.

Finally, agent $\mathfrak{m}$ checks whether the hypotheses in its hypothetical mutable belief base are still consistent with its core beliefs and its revised factual information. If so, it does not modify them. If not, it minimally contracts its hypothetical mutable belief base: it takes the intersection of the subsets of the hypothetical mutable belief base which are maximally consistent with respect to the core belief base and its revised factual information.

For notational convenience, we write $Rev^{core}(\Sigma_c, \Sigma_h, \Sigma_f)$ to denote Σ'_c, $Rev^{hyp}(\Sigma_c, \Sigma_h, \Sigma_f)$ to denote Σ'_h and $Rev^{fact}(\Sigma_c, \Sigma_h, \Sigma_f)$ to denote Σ'_f. Note that if $\Sigma = \Sigma_c \cup \Sigma_h \cup \Sigma_f$ is propositionally consistent, then $Rev^{core}(\Sigma_c, \Sigma_h, \Sigma_f) \cup Rev^{hyp}(\Sigma_c, \Sigma_h, \Sigma_f) \cup Rev^{fact}(\Sigma_c, \Sigma_h, \Sigma_f)$ is propositionally consistent too.

In the next section, we will provide a formalization of the Yōkai board-game with the aid of the language $\mathcal{L}^+(ATM)$. We will represent agent $\mathfrak{m}$'s action repertoire to be used in the action selection problem as a set of events in $Act_{\mathfrak{m}}$ affecting $\mathfrak{m}$'s beliefs. We assume in its turn agent $\mathfrak{m}$ faces four consecutive action selection problems. First, agent $\mathfrak{m}$ has to decide at which card to look. It faces this problem twice. Then, it has to decide which card to move and to which position. Finally, it has to decide whether to activate a hint or mark a card with a hint. For every action of $\mathfrak{m}$, we will specify the corresponding executability precondition.

Moreover, we will specify agent $\mathfrak{m}$'s available information at the beginning of the game and clearly distinguish mutable (factual and hypothetical) information from core information.

5. Game modeling

In this section, we first formalize all the static aspects of the game (the different rules, the initial state of the game, etc.) and then the action representation. Finally, we present an example of action selection by agent $\mathfrak{m}$ in the game.

5.1. Static aspects

Let be the following sets:

$$TIME = \{0, 1, .., end\} \text{ where } end = 56,$$
$$TIME^* = TIME \setminus \{0\},$$
$$GRID = \{1, \ldots, 32\} \times \{1, \ldots, 32\},$$
$$IPOS = \big\{(l, c) \in GRID : l, c \in \{15, ..., 18\}\big\},$$
$$COLORS = \{\mathrm{r}, \mathrm{g}, \mathrm{b}, \mathrm{y}\},$$
$$HINTS = 2^{COLORS} \setminus \big\{\{\}, \{\mathrm{r}, \mathrm{g}, \mathrm{b}, \mathrm{y}\}\big\},$$
$$CARDS = \{1, \ldots, 16\}$$
$$CARDS^n = \{X \in 2^{CARDS} : |X| = n\} \text{ with } n \in \mathbb{N}$$

There are seven hints at the start of the game, and each player must take turns activating a hint or using a hint to mark a card. Thus, each player plays 7 times. As each player must take 4 different actions in turn, the game lasts a maximum of $(7 \times 4) \times 2 = 56$ time units (56 actions are executed during a game). So, let *TIME* be the set of time points, including initial state of the game at time 0. End of the game is marked by the natural $end \in TIME$ and $end = 56$.

However, each player only performs the action of moving a card once among the 4 actions she/it must perform during her/its turn. There are therefore a total of $7 \times 2 = 14$ card moves. As in the initial state the 16 cards are placed in a square of 4×4 cards, whatever the movements made during a game, the cards will evolve in a grid of 32×32 positions represented by the set *GRID*, and *IPOS* represents the set of cards positions at the start of the game.

A hint is viewed as a non empty subset of 1, 2 or 3 colors among 4 different colors (r for red, g for green, b for blue and y for yellow). *HINTS* is the set of all hints and $|HINTS| = 14$. When the game starts, only 7 hints are available.

Vocabulary. The set of atomic proposition *Atm* is defined as follows:

$$ATM = \bigcup_{\substack{t\in TIME\\ x\in CARDS\\ c\in COLORS}} \{col^t_{x,c}\} \cup \bigcup_{\substack{t\in TIME\\ x\in CARDS\\ p\in GRID}} \{pos^t_{x,p}\} \cup \bigcup_{\substack{t\in TIME\\ h\in HINTS}} \{active^t_h\} \cup \bigcup_{\substack{t\in TIME\\ x\in CARDS\\ h\in HINTS}} \{mark^t_{x,h}\} \cup \bigcup_{\substack{t\in TIME\\ x\in CARDS\\ p\in GRID}} \{legMov^t_{x,p}\}$$

where: $col^t_{x,c}$ is true iff card x is of color c at time t; $pos^t_{x,p}$ is true iff card x is at position p at time t; $active^t_h$ is true iff hint h is enabled at time t; $mark^t_{x,h}$ is true iff card x is marked with hint h at time t. Finally, $legMov^t_{x,p}$ is true iff to move card x to position p at time t is legal (see paragraph "The separation constraint" page 14 for more details).

Moreover, we define a function $\sigma : CARDS \longrightarrow IPOS$ that assigns to each card x a position p in the initial positions set, and a function $NEIG : GRID \longrightarrow 2^{GRID}$ that assigns to each position (l,c) the set of its neighboring positions in the grid $NEIG\big((l,c)\big) = \big\{(l+1,c),(l-1,c),(l,c+1),(l,c-1)\big\} \cap GRID$.

In the following, we consider that belief bases of agents are defined in $\mathcal{L}_0$ and are divided into two (sub)bases: the *core beliefs* base (containing all the beliefs that cannot be modified during the game), and the *mutable beliefs* base (containing all the belief that may change). As we want to model the game from $\mathfrak{m}$'s perspective, we only model its beliefs about the facts of the game or about the beliefs of agent $\mathfrak{h}$.

Initial mutable beliefs of agent $\mathfrak{m}$. Recall (see Section 4.2) that mutable beliefs base $\Sigma_m = \Sigma_f \cup \Sigma_h$ (union of factual beliefs base and hypothetical beliefs base).

In initial state, Σ_f (the factual beliefs base of agent $\mathfrak{m}$) contains not only beliefs on facts of the world (which is captured by the definition of the set Σ^W_f) but also on the beliefs of the agent $\mathfrak{h}$ on these facts of the world (see below the definition of $\Sigma^{\mathfrak{h}}_f$).

Σ^W_f contains all $\mathfrak{m}$'s beliefs about the initial state of the game. It includes the fact that: time is at least 0, each card position is known (we assume that function σ is known), the cards color is not known, only seven hints are available et and the others are not known, none is activated yet, and none marks a card yet. Moreover, agent $\mathfrak{m}$ knows all positions that are adjacent and those that are not.

$\Sigma^{\mathfrak{h}}_f$ includes the fact that agent $\mathfrak{m}$ believes that the initial state of the game is also known by agent $\mathfrak{h}$. That is, m believes that $\mathfrak{h}$ believes all the facts included in Σ^W_f, plus the fact that $\mathfrak{h}$ does not know the color of any card.

Finally, Σ_f is the union of Σ^W_f and $\Sigma^{\mathfrak{h}}_f$. So, formally:

$$\Sigma^W_f = \{now^{\geqslant 0}\} \cup \bigcup_{x\in CARDS} \{pos^0_{x,\sigma(x)}\} \cup \bigcup_{\substack{x\in CARDS\\ c\in COLORS}} \{\neg \triangle_{\mathfrak{m}}\, col^0_{x,c}\} \cup \bigcup_{h\in HINTS} \{\neg active^0_h\} \cup \bigcup_{\substack{x\in CARDS\\ h\in HINTS}} \{\neg mark^0_{x,h}\}$$

$$\Sigma^{\mathfrak{h}}_f = \bigcup_{\alpha\in\Sigma^W_f} \{\triangle^0_{\mathfrak{h}}\,\alpha\} \cup \bigcup_{\substack{x\in CARDS\\ c\in COLORS}} \{\neg \triangle^0_{\mathfrak{h}}\, col^0_{x,c}\}$$

$$\Sigma_f = \Sigma^W_f \cup \Sigma^{\mathfrak{h}}_f$$

Note that Σ_f does not contain any conjunction or disjunction big operator. It is for technical considerations: when expanding the belief base of the agent following the execution of an action, formulas can be deleted from the base. If these are present in the form of a conjunction, it suffices that one term of this conjunction is inconsistent with the new information for the whole conjunction to be deleted. In order to minimize the suppressed information, each conjunction is seen as a set of (sub-)formulas.

In initial state, the hypothetical beliefs base Σ_h of agent $\mathfrak{m}$ includes the fact that $\mathfrak{m}$ believes that when a card is marked with a hint that provides a set of possible colors for that card, this card cannot have a color other than those provided by the hint. We could suppose that agent $\mathfrak{m}$ believes that agent $\mathfrak{h}$ has the same beliefs, but as we do not currently use such beliefs in $\mathfrak{m}$'s reasoning, we omit here to add this kind of beliefs to Σ_h. That is:

$$\Sigma_h = \bigcup_{\substack{x\in CARDS\\ h\in HINTS\\ c\in COLORS\setminus h}} \{mark^0_{x,h} \rightarrow \neg col^0_{x,c}\ ,\ \triangle^0_{\mathfrak{h}}\, mark^0_{x,h} \rightarrow \triangle^0_{\mathfrak{h}}\, \neg col^0_{x,c}\}$$

For example, $\Sigma_h \supseteq \{mark^0_{1,\{g,r\}} \rightarrow \neg col^0_{1,b}, mark^0_{1,\{g,r\}} \rightarrow \neg col^0_{1,y}\}$. Suppose now that $mark^0_{1,\{g,r\}} \in \Sigma_f$, then $\Sigma_c \cup \Sigma_m \models \Box_{\mathfrak{m}} \left((col^0_{1,g} \vee col^0_{1,r}) \wedge \neg col^0_{1,b} \wedge \neg col^0_{1,y}\right)$ holds (see below for Σ_c definition).

Why is this type of knowledge in Σ_h and not in Σ_f? Because it is general knowledge that can be questioned (like default rules of reasoning). Thus, if we suppose that agent $\mathfrak{h}$ puts a hint $h = \{g\}$ on a card (which means "this card is green") then agent $\mathfrak{m}$ can deduce implicitly (thanks to Σ_h) that this card is green. But in fact, four cases are possible: 1) $\mathfrak{m}$ already explicitly believes that the card is green, in which case $\mathfrak{m}$ does not learn anything that it did not already know; 2) $\mathfrak{m}$ does not know the color of the card, and in this case it can implicitly deduce that the card is neither yellow, nor red, nor blue; 3) $\mathfrak{m}$ already explicitly believes that the card is of another color (red, for example), and in this case it is primarily Σ_h that will be revised; 4) $\mathfrak{m}$ implicitly believes that the card cannot be green (because $\mathfrak{m}$ already knows the 4 green cards for example), and then this case is treated as 3).

Initial core beliefs of agent $\mathfrak{m}$. Core belief base includes all the agent $\mathfrak{m}$'s beliefs that cannot change during the game. It concerns both integrity constraints (that describe rules of the game) and successor state axioms (SSAs) that describe the building of the belief base after the execution of an action. SSAs describe both what changes in the new state, and what do not (thanks to frame axioms). So, we define the following integrity constraints (IC):

$$\Sigma^{ic}_c = \bigcup_{t \in TIME} \Bigg\{ \bigwedge_{x \in CARDS} \bigvee_{p \in GRID} pos^t_{x,p}\,, \tag{ICP1}$$

$$\bigwedge_{\substack{x \in CARDS \\ p,p' \in GRID : p \neq p'}} \neg(pos^t_{x,p} \wedge pos^t_{x,p'})\,, \tag{ICP2}$$

$$\bigwedge_{\substack{p \in GRID \\ x,x' \in CARDS : x \neq x'}} \neg(pos^t_{x,p} \wedge pos^t_{x',p})\,, \tag{ICP3}$$

$$\bigwedge_{x \in CARDS} \bigvee_{c \in COLORS} col^t_{x,c}\,, \tag{ICC4}$$

$$\bigwedge_{\substack{x \in CARDS \\ c,c' \in COLORS : c \neq c'}} \neg(col^t_{x,c} \wedge col^t_{x,c'})\,, \tag{ICC5}$$

$$\bigwedge_{c \in COLORS} \bigvee_{X \in CARDS^4} \Big(\bigwedge_{x \in X} col^t_{x,c} \wedge \bigwedge_{x \notin X} \neg col^t_{x,c} \Big)\,, \tag{ICC6}$$

$$\bigwedge_{\substack{h \in HINTS \\ x,x' \in CARDS \\ x \neq x'}} \neg(mark^t_{x,h} \wedge mark^t_{x',h})\,, \tag{ICH7}$$

$$\bigwedge_{\substack{x \in CARDS \\ h \in HINTS}} \neg(active^t_h \wedge mark^t_{x,h}) \Bigg\} \tag{ICH8}$$

ICP1 and ICP2 mean that each card has at least one, and at most one, position respectively ; ICP3 means that each position can only accommodate one card. In the same way, ICC4 and ICC5 means that each card has at least one, and at most one, color respectively. ICC6 means that there are exactly 4 cards of each color. ICH7 means a hint cannot marks two different cards. ICH8 means that a hint is either not active and not marking, active and not marking, or not active and marking. Finally, Σ_c assumes that agent $\mathfrak{m}$ believes that the initial state of the game is also known by agent $\mathfrak{h}$.

Note that by definition, Σ_c cannot be revised by any action, one can add conjunctions directly to this set (rather than the set of their sub-formulas).

Frame axioms (FA) describe the facts that does not change after the execution of an action. For convenience, we define the following FA abbreviations (for every $X \subseteq CARDS$, $H \subseteq HINTS$ and $t \in TIME^*$):

$$\mathsf{posFA}^t_X \stackrel{\text{def}}{=} \bigwedge_{\substack{x \in CARDS \setminus X \\ p \in GRID}} (pos^t_{x,p} \leftrightarrow pos^{t-1}_{x,p})$$

$$\mathsf{colFA}^t_X \stackrel{\text{def}}{=} \bigwedge_{\substack{x \in CARDS \setminus X \\ c \in COLORS}} \Big((col^t_{x,c} \leftrightarrow col^{t-1}_{x,c}) \wedge (\neg \triangle_{\mathfrak{m}}\, col^t_{x,c} \leftrightarrow \neg \triangle_{\mathfrak{m}}\, col^{t-1}_{x,c}) \Big)$$

$$\mathsf{hintFA}^t_H \stackrel{\text{def}}{=} \bigwedge_{h \in HINTS \setminus H} (active^t_h \leftrightarrow active^{t-1}_h) \wedge \bigwedge_{\substack{x \in CARDS \\ h \in HINTS \setminus H}} (mark^t_{x,h} \leftrightarrow mark^{t-1}_{x,h})$$

posFA^t_X (resp. colFA^t_X) reads "the position (resp. color) of every cards except those in X is preserved from time $t-1$ to t". $\mathsf{hintFA}^t_X H$ reads "the status (active or marking) of every hints except those in H is preserved from time $t-1$ to t".

Finally, we define the following successor state axioms that are useful for planning since they allow to deduce what will be true at some time t in the future:

$$\Sigma^{ssa}_c = \bigcup_{t \in TIME^*} \Bigg\{$$

$$\bigwedge_{\substack{x \in CARDS \\ p \in GRID}} (pos^t_{x,p} \wedge \neg pos^{t-1}_{x,p} \rightarrow \mathsf{posFA}^t_{\{x\}} \wedge \mathsf{colFA}^t_{\emptyset} \wedge \mathsf{hintFA}^t_{\emptyset}), \tag{SSA9}$$

$$\bigwedge_{\substack{x \in CARDS \\ c \in COLORS}} (col^t_{x,c} \wedge \neg \triangle_{\mathfrak{m}}\, col^{t-1}_{x,c} \rightarrow \mathsf{posFA}^t_{\emptyset} \wedge \mathsf{colFA}^t_{\{x\}} \wedge \mathsf{hintFA}^t_{\emptyset}), \tag{SSA10}$$

$$\bigwedge_{h \in HINTS} (active^t_h \wedge \neg active^{t-1}_h \rightarrow \mathsf{posFA}^t_{\emptyset} \wedge \mathsf{colFA}^t_{\emptyset} \wedge \mathsf{hintFA}^t_{\{h\}}), \tag{SSA11}$$

$$\bigwedge_{\substack{x \in CARDS \\ h \in HINTS}} (mark^t_{x,h} \wedge \neg mark^{t-1}_{x,h} \rightarrow \mathsf{posFA}^t_{\emptyset} \wedge \mathsf{colFA}^t_{\emptyset} \wedge \mathsf{hintFA}^t_{\{h\}}) \tag{SSA12}$$

$$\Bigg\}$$

SSA9 means that if card x has a new position p at time t, the position of others cards, the color of every cards, and the status of all hints, remain unchanged from $t-1$ to t. SSA10 means that if card x has a new color c at time t, the position of every cards, the color of every other cards, and the status of all hints, remain unchanged from $t-1$ to t. SSA11 and SSA12 mean that if a hint becomes active (resp. marks a card) between time $t-1$ and t then neither positions and colors of cards did change nor the status of other hints. Moreover, we assume that at least one action is performed at each time point. Formally, this is represented by the disjunction of antecedents of SSA9 to SSA12.

Finally, we suppose that $\mathfrak{m}$ believes that all the fact in its core belief base are also known by the other agent $\mathfrak{h}$.

$$\Sigma_c = \Sigma^{ic}_c \cup \Sigma^{ssa}_c$$

In principle, agent $\mathfrak{m}$ believes that agent $\mathfrak{h}$ shares the same rules of the game (set Σ^{ic}_c) and the same successor states axioms (set Σ^{ssa}_c), but for the sake of simplicity, we omit this part which is not currently used by $\mathfrak{m}$ for planning actions.

5.2. Dynamic aspects

We introduce in what follows actions of perceptions. These actions allow the agent $\mathfrak{m}$ to expand its belief base by a certain formula.

Vocabulary. For convenience, we define the action repertoire $ACT \subseteq EVT$ of agent $\mathfrak{m}$ (for every $t < end$, $x \in CARDS$, $p \in GRID$, $h \in HINTS$):

$$+_{\mathfrak{m}}^{\mathsf{col}_{x,c}^{t+1}} \stackrel{\text{def}}{=} +_{\mathfrak{m}}^{t+1}\Big(col_{x,c}^{t+1} \wedge \triangle_{\mathfrak{m}}\, col_{x,c}^{t+1} \wedge (\triangle_{\mathfrak{h}}^{t+1} \bigvee_{c' \in COLORS} \triangle_{\mathfrak{m}}\, col_{x,c'}^{t+1}) \wedge \triangle_{\mathfrak{h}}^{t+1}\, now^{\geqslant t+1}\Big)$$

$$+_{\mathfrak{m}}^{\mathsf{pos}_{x,p}^{t+1}} \stackrel{\text{def}}{=} +_{\mathfrak{m}}^{t+1}\Big(pos_{x,p}^{t+1} \wedge \triangle_{\mathfrak{m}}\, pos_{x,p}^{t+1} \wedge \triangle_{\mathfrak{h}}^{t+1}\, pos_{x,p}^{t+1} \wedge \triangle_{\mathfrak{h}}^{t+1}\, now^{\geqslant t+1}\Big)$$

$$+_{\mathfrak{m}}^{\mathsf{actHint}_{h}^{t+1}} \stackrel{\text{def}}{=} +_{\mathfrak{m}}^{t+1}\Big(active_{h}^{t+1} \wedge \triangle_{\mathfrak{m}}\, active_{h}^{t+1} \wedge \triangle_{\mathfrak{h}}^{t+1}\, active_{h}^{t+1} \wedge \triangle_{\mathfrak{h}}^{t+1}\, now^{\geqslant t+1}\Big)$$

$$+_{\mathfrak{m}}^{\mathsf{markHint}_{x,h}^{t+1}} \stackrel{\text{def}}{=} +_{\mathfrak{m}}^{t+1}\Big(mark_{x,h}^{t+1} \wedge \triangle_{\mathfrak{m}}\, mark_{x,h}^{t+1} \wedge \triangle_{\mathfrak{h}}^{t+1}\, mark_{x,h}^{t+1} \wedge \triangle_{\mathfrak{h}}^{t+1}\, now^{\geqslant t+1}\Big)$$

As each action entails a new time point, each action execution entails the fact that the agent learns not only that time increases (which is taken into account directly in the semantics, see Definition 1), but also that time increases for agent $\mathfrak{h}$ (so, $\triangle_{\mathfrak{h}}^{t+1}\, now^{\geqslant t+1}$ must be added to $\mathfrak{m}$'s belief base). Moreover, each time p^t is added, $\triangle_{\mathfrak{m}}\, p^t$ is also added. The reason is that in the case where the agent does not explicitly believe that p ($\neg \triangle_{\mathfrak{m}}\, p^t$) and we want to add p^t to the belief base of the agent, we have to remove $\neg \triangle_{\mathfrak{m}}\, p^t$ from the base. For example, in the initial state, the agent does not know the color of the card x (that is, for any color c, $\neg \triangle_{\mathfrak{m}}\, col_{x,c}^0$ belongs to the base). As soon as he observes the color of a card (which is green, for example), then $\neg \triangle_{\mathfrak{m}}\, col_{x,\mathrm{g}}^1$ must be deleted from the base, which will be done automatically by our revision module as soon as $\triangle_{\mathfrak{m}}\, col_{x,\mathrm{g}}^1$ is added to the database (in addition to $col_{x,\mathrm{g}}^1$).

Note also that $\triangle_{\mathfrak{h}}^{t+1} \bigvee_{c' \in COLORS} \triangle_{\mathfrak{m}}\, col_{x,c'}^{t+1}$ reads "agent $\mathfrak{h}$ believes that $\mathfrak{m}$ knows the color of card c'", that is quite different from $\triangle_{\mathfrak{h}}^{t+1} \triangle_{\mathfrak{m}} \bigvee_{c' \in COLORS} col_{x,c'}^{t+1}$ that reads "agent $\mathfrak{h}$ believes that agent $\mathfrak{m}$ believes that card x has at least one color (among *COLORS*)".

Moreover, each action about a fact also informs agent $\mathfrak{m}$ that $\mathfrak{h}$ believes this fact.

Finally, note that these actions concern the point of view of agent $\mathfrak{m}$ but say nothing about their author (which can be $\mathfrak{m}$ or $\mathfrak{h}$).

The separation constraint. When moving a card from a position p to a position p', it is forbidden to create two separate groups of cards (see Figure 1b). In other words, there must be a path between a given card x and all the other cards y, which automatically ensures that there is a path between any two different cards y' and y'' of the game.

We have a logical characterization of this constraint, but it is complex and for the sake of simplicity and for efficiency, we introduced in *ATM* definition (see paragraph "Vocabulary" page 11) an atomic formula $legMov_{x,p}^t$ whose truth value is supposed to be updated at a metalogical level.

The meaning of "$legMov_{x,p}^t$ is true" is: "the move of card x towards position p at time t is authorized by the rules of the game", that is: p is currently an empty position and there is a sequence of adjacent positions between p and any other occupied positions (except the initial position p'' of card x) through the set of currently occupied positions (excluding p'').

Action preconditions. We assume now that the operators in *ACT* have the following executability preconditions, for $t \in TIME$ such that $t < end$, $x \in CARDS$, $c \in COLORS$, $h \in HINTS$ and $now^{=t} \stackrel{\text{def}}{=} now^{\geqslant t} \wedge \neg now^{\geqslant t+1}$:

$$\mathcal{P}(+_{\mathfrak{m}}^{\mathsf{col}_{x,c}^{t+1}}) \stackrel{\text{def}}{=} now^{=t} \wedge \Box_{\mathfrak{m}} \bigwedge_{h \in HINTS} \neg mark_{x,h}^t$$

$$\mathcal{P}(+_{\mathfrak{m}}^{\mathsf{pos}_{x,p}^{t+1}}) \stackrel{\text{def}}{=} now^{=t} \wedge \Box_{\mathfrak{m}} \bigwedge_{h \in HINTS} \neg mark_{x,h}^t \wedge \Box_{\mathfrak{m}}\, legMov_{x,p}^t$$

$$\mathcal{P}(+_{\mathfrak{m}}^{\mathsf{actHint}_h^{t+1}}) \quad \stackrel{\text{def}}{=} \quad now^{=t}$$

$$\mathcal{P}(+_{\mathfrak{m}}^{\mathsf{markHint}_{x,h}^{t+1}}) \quad \stackrel{\text{def}}{=} \quad now^{=t} \wedge \Box_{\mathfrak{m}}\, active_h^t \wedge \bigvee_{c\in h} \Box_{\mathfrak{m}}\, col_{x,c}^t \wedge \Box_{\mathfrak{m}} \bigwedge_{\substack{h'\in HINTS\\ h'\neq h}} \neg mark_{x,h'}^t$$

$\mathcal{P}(+_{\mathfrak{m}}^{\mathsf{col}_{x,c}^{t+1}})$ reads "agent $\mathfrak{m}$ implicitly believes that time is currently equal to t and also implicitly believes that no hint marks card x"; $\mathcal{P}(+_{\mathfrak{m}}^{\mathsf{pos}_{x,p}^{t+1}})$ reads "agent $\mathfrak{m}$ implicitly believes that time is currently equal to t and also implicitly believes that no hint marks card x and that it is currently authorized, w.r.t. rules of the game, to move cards x to position p"; $\mathcal{P}(+_{\mathfrak{m}}^{\mathsf{actHint}_h^{t+1}})$ reads "agent $\mathfrak{m}$ implicitly believes that time is currently equal to t"; $\mathcal{P}(+_{\mathfrak{m}}^{\mathsf{markHint}_{x,h}^{t+1}})$ reads "agent $\mathfrak{m}$ implicitly believes that time is currently equal to t, that hint h is currently active, that h includes the colors of card x, and that no other hint h' already marks card x".

5.3. Example of action selection

In this section, we are going to illustrate the action selection problem defined in Section 4.1 on a specific configuration of the game. We assume is agent $\mathfrak{m}$'s turn to play and its goal α_G is to mark a card which is not actually marked, whose color is not known by the human and which is adjacent to a card of the same color. That is,

$$\alpha_G \quad \stackrel{\text{def}}{=} \quad \bigvee_{\substack{x\in CARDS\\ p\in GRID\\ c\in COLORS}} \Big(pos_{x,p}^t \wedge col_{x,c}^t \wedge \neg \triangle_{\mathfrak{h}}^t\, col_{x,c}^t \wedge \bigwedge_{h\in HINTS} \neg mark_{x,h}^t \wedge \bigvee_{\substack{p'\in NEIG(p)\\ x'\in CARDS}} (pos_{x',p'}^t \wedge col_{x',c}^t) \wedge \bigvee_{\substack{h\in HINTS\\ c\in h}} mark_{x,h}^{t+1} \Big)$$

We suppose agent $\mathfrak{m}$ has the following information in its factual mutable belief base Σ_f:

$$\Sigma_f \supseteq \big\{ active_{\{\mathrm{r,g}\}}^t, pos_{3,(16,16)}^t, pos_{6,(16,17)}^t, col_{3,\mathrm{r}}^t, col_{6,\mathrm{r}}^t, \neg \triangle_{\mathfrak{h}}^t\, col_{3,\mathrm{r}}^t \big\} \cup \bigcup_{h\in HINTS} \big\{ \neg mark_{3,h}^t \big\} \cup \bigcup_{t'\leqslant t} \big\{ now^{\geqslant t'} \big\}$$

Moreover, for every $t'' > t$, we suppose that $now^{\geqslant t''} \notin \Sigma = (\Sigma_m \cup \Sigma_c)$. This means that agent $\mathfrak{m}$ knows that the current time is exactly t.

Agent $\mathfrak{m}$ is at the last step of its turn. It has decide how to mark one card with an active hint. Thus, its action repertoire Op includes all and only actions of type $+_{\mathfrak{m}}^{\mathsf{markHint}_{x,h}^t}$. It is straightforward to verify that action $+_{\mathfrak{m}}^{\mathsf{markHint}_{3,\{\mathrm{r,g}\}}^t}$ of marking card 3 with the hint $\{\mathrm{r,g}\}$ is a solution to the action selection problem $\langle \Sigma, Op, \alpha_G \rangle$ so defined. Indeed, according to the available information in agent $\mathfrak{m}$'s belief base Σ, card 3 at position $(16,16)$ is unmarked and is adjacent to card 6 at position $(16,17)$, the two cards are both red and the color of card 3 is not known by agent $\mathfrak{h}$.

Appendix A presents the modeling of some goals. The system designer can of course write additional goals or replace those presented by others. (The goal is just to show that the language can model simple or more complex goals.) Parameters can also be introduced in order to optimize goal planning. In this case, an algorithm describes how to calculate these parameters.

6. Implementation and experiments

Implementation. The architecture presented in Figure 2 works as an integrated system[5]. All the processes, interfaces and exchange of data between the modules work according to the game rules.

[5] https://github.com/iritlab/yokai

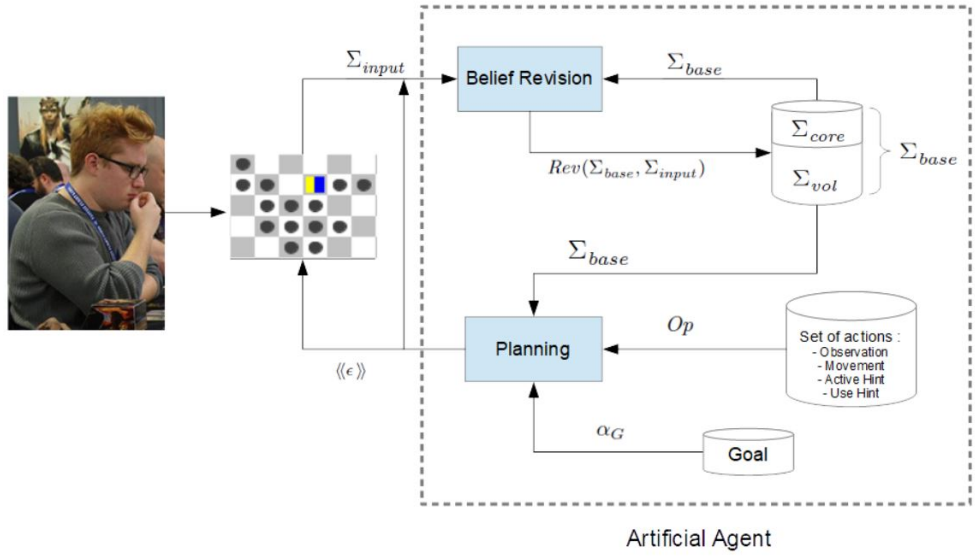

Figure 2. System architecture

The GUI module allows players to perform observations, movements, and activate or mark a card with a hint. The activation of hints is randomly generated correctly and according to the specifications for hints detailed in the paragraph "hints" In Section 2. In addition, the generation of the set of data after each game exactly reflects the current state of the game (current position, marked cards, occupied positions, perimeter of the set of cards, etc.).

Some data can be computed inside the logical model (e.g., a position in the grid is occupied if there is a card occupying this position). But the fact of having to recompute them each time the solver is requested, may be very costly in term of computation time and memory space. Thus, the graphical interface manages a certain amount of data natively (position of the cards, movement authorized or not, etc.), and it can be categorized as a *third agent*, which controls, for example, the real colors of the cards as well as the hints generation.

It is important to mention that the GUI will verify when the players manage to group the cards, because it has perfect information of the distribution of all cards in the grid.

The belief revision module can drop the older beliefs that contradict the new inputs. Similarly, the hierarchy of goals works fine and manages to group the cards according to the cards known by agent $\mathfrak{m}$.

Data set. Our experiment was constructed as follows. We conducted 10 *experiment sets*. Each experiment set consisted of 20 *games*: 10 games were played by the human-machine *configuration type* ($\mathfrak{h}$-$\mathfrak{m}$) and 10 were played by the human-human configuration type ($\mathfrak{h}$-$\mathfrak{h}$). Finally, the experiment includes 200 games of Yōkai (10 times 20 games). Every time a new game started, the system generated a random distribution of the 16 cards in the grid.

Collaboration level analysis. First, we are interested in measuring the *level of collaboration* between players, regardless of whether players win or fail the game. We decided to compute the level of collaboration on the basis of the number of *y-groups* (Yōkai groups) when the game is over. We call "y-group" a group of cards of the same color without separation for this group. If k is the size of a y-group (that is, it contains k cards), we write it is a k-y-group. Then, the *collaboration level* is n when, at the end of the game, players have built n 4-y-groups.

For instance, consider the example shown in Figure 3 above. In this figure, although in both cases the players lost the game, it is easy to see that the collaboration between players was more efficient in **case 1** than in **case 2**: since in **case 1** we observe three 4-y-groups while in **case 2** we do not observe any 4-y-group, the collaboration level is 3 in **case 1** and 0 in **case 2**.

The players win the game when they build four 4-y-groups. Therefore, the collaboration level is maximal in the winning situation (i.e., it is equal to 4).

In the following, we seek to measure the collaboration level in different cases.

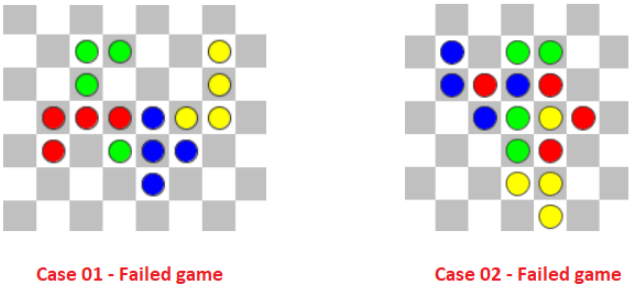

Figure 3. Two different scenarios when players lost the game

We show in Table 1 the results achieved during the games performed in the first experiment set. Note that the maximal score for a game configuration in an experiment set is 40 (that is, 10 times 4 points).

Table 1
Games performed during *Experiment set 1*

Game	ɧ-ɱ		ɧ-ɧ	
	# 4-y-groups	**Score**	**# 4-y-groups**	**Score**
1	1	-	1	-
2	1	-	2	-
3	4	8	2	-
4	1	-	2	-
5	3	-	1	-
6	4	8	4	9
7	4	7	3	-
8	2	-	2	-
9	1	-	1	-
10	1	-	0	-
Total	22		18	

The entire data set of the experiment is presented in Table 2 and is plotted in Figure 4.
In Table 2, the average score of the ɧ-ɱ configuration is 26.7, and that of the ɧ-ɧ configuration is 21.6.

Table 2
Average total points by experiment set

Configuration type	**Experiment set**									
	1	2	3	4	5	6	7	8	9	10
ɧ-ɱ	22	28	33	32	28	30	19	26	23	26
ɧ-ɧ	18	23	21	23	24	25	17	19	21	25

In Figure 4, we can see that the plot representing the ɧ-ɱ configuration (in blue) is always located above that representing the ɧ-ɧ configuration (in red in the figure). This means that the level of collaboration is always higher in the first type than in the second type. Of course, one can object that a very high level of collaboration can still lead to a low result, but remember that here we define the level of collaboration according to the number of 4-y-groups which is an indicator based on the results of the games.

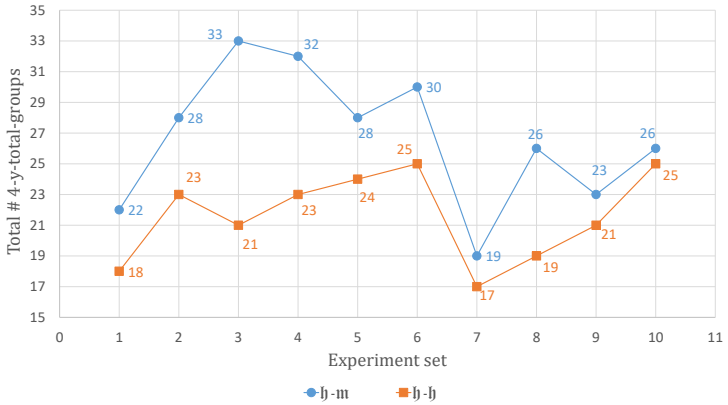

Figure 4. Total # 4-y-groups by experiment set

Score analysis. The next step was to count the number of won games among the ten games performed by type of configuration in each experiment set. Even more, we are interested in the highest score achieved among the ten games performed by type of configuration. Recall the score is computed according to the formula detailed in Section 2. We represented both measures in Table 3.

Table 3
Games won and best scores for each experiment set

Experiment set	ɦ-m		ɦ-ɦ	
	Games won	**Best score**	**Games won**	**Best score**
1	3	8	1	9
2	3	9	2	7
3	5	9	2	8
4	4	11	1	7
5	2	12	1	5
6	4	7	3	8
7	2	8	0	0
8	2	7	1	7
9	1	7	0	0
10	2	8	2	9

We show in Figure 5 the number of games won for each type of configuration. Note that the best result by set is 10. Again, we can see that the number of games won in blue (ɦ-m) is always greater than or equal to that in red (ɦ-ɦ). The best result is 5 (that is, one out of two games was won within this set) and is obtained by ɦ-m collaboration. For ɦ-ɦ collaboration, the best score is only 3.

In Figure 6 we show the best score achieved in each experiment set and for each type of collaboration. We can observe that in three experiment sets the ɦ-m configuration was able to overcome the score achieved by the ɦ-ɦ configuration.

Discussion. Let's first compare the level of collaboration according to the type of game configuration. We can see in Figure 4 that the ɦ-m collaboration overcomes the ɦ-ɦ collaboration in grouping cards of the same color. Among the 10 experiment sets, the ɦ-ɦ configuration was close to reaching the same level of collaboration as the ɦ-m configuration in only two sets (7 and 9) .

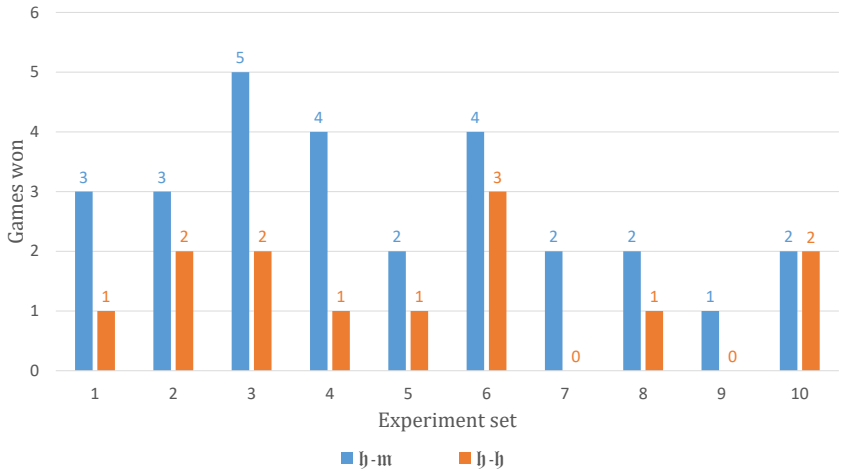

Figure 5. Number of games won in each experimental set by each game configuration

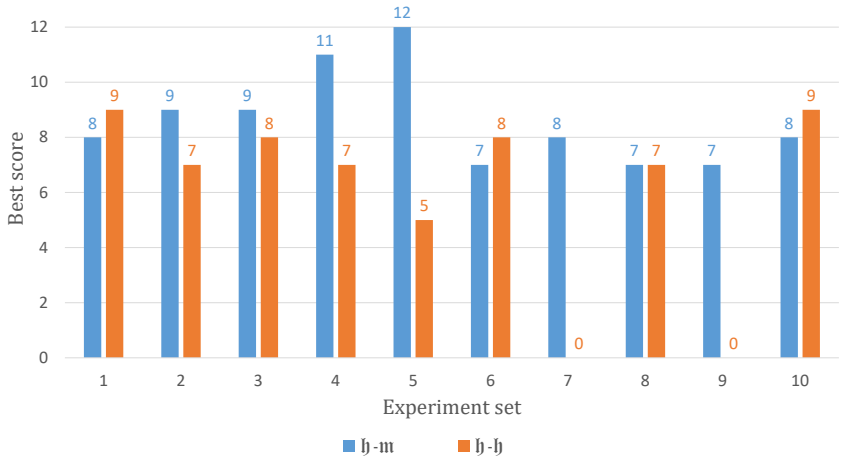

Figure 6. Best score achieved among games performed in each experimental set by each game configuration

Additionally, analysis of the scoring system from Figure 5 shows that the number of games won in the configuration ɧ-ɱ is higher than that of the configuration ɧ-ɧ in all experiment sets. Moreover, Figure 6 shows that the highest score obtained in the configuration ɧ-ɱ is on average better than that in the configuration ɧ-ɧ, since there are only three cases (sets 1, 6 and 10) where the configuration ɧ-ɧ exceeds the best score achieved in the configuration ɧ-ɱ.

By analyzing the games played during the experiment, it seems that if the hints activated at the beginning of the game are those with a single color, then there is a greater probability of winning the game because one player can use these hints to guide the other player about the cards it knows and the other doesn't. Similarly, marking cards with a hint having three colors sometimes makes it possible to deduce the color of a card according to the other active hints. For example, suppose agent ɧ observes a card while there are three active hints $\{y, b\}$, $\{y, r\}$ and $\{y, g, r\}$. Suppose also that a player marks a card with the hint $\{y, g, r\}$. In such a case, and assuming that we keep in mind that the smaller hints are played first because they are more informative, we could deduce that the marked card is green. In order not to complicate our model, this type of reasoning has not been taken into account for the moment. Nevertheless, it is not a technical impossibility: as we did for the rules of the type "if a card is marked with a hint h then its color cannot be a color not belonging to h", we could add similar rules to the Σ_h set of revisable beliefs

of the agent $\mathfrak{m}$ so that we can remove them if necessary. For example, suppose that using such a rule, the agent $\mathfrak{m}$ deduces that the marked card is green, and suppose that later, by observing other cards, $\mathfrak{m}$ actually knows the 4 cards that are really green. $\mathfrak{m}$ should no longer deduce from the clue that the marked card is green and $\mathfrak{m}$ just has to revise its beliefs by removing the rule allowing it to deduce the color of the marked card.

Following the Yōkai rules, we can only activate a hint or use it for marking a card. Currently, the algorithm that manages the fact of marking a card using a hint (see Algorithm 4) only allows activating a new hint if no more hints are playable or available. Nevertheless, sometimes it is better to activate a new hint than to use a hint in a non-informative way for the other player. For example, this case can occur if the only active hint is $\{\mathrm{g}\}$ and I know that the other player already knows all the green cards that I know. The algorithm can be improved, although the idea was not in this work to describe all the possible goals and the best strategy to seek to satisfy them, but rather to simply show how to proceed, and the capacity of the formal language to do this work.

7. Conclusion

We introduced a simple epistemic language to represent the knowledge and actions of an artificial player in the context of the cooperative board game Yōkai. We have shown that this game requires a combination of theory of mind (ToM) and temporal and spatial reasoning to be played effectively by an artificial agent. Our approach relies on SAT given the existence of a polysize satisfiability preserving translation of the epistemic language into propositional logic.

We have shown that there are mainly two factors that facilitate the implementation: the process of updating beliefs and the fact of delegating certain validations to the graphical interface.

We have also shown that the dynamic generation of actions improves the performance of the planner. This improvement is based on the fact that at each moment the machine performs only one action, and that the type of this action is known in advance (thanks to the sequence of actions in the same turn). For example, if the current action is in fact the machine's third action, then it is of type 'move a card'. Thus, it is not efficient to carry out a planning with a repertoire of actions which considers all the types of possible operations. Thus, if the action the machine must perform is to move a card, then the system only generates move actions in the actions directory. Additionally, the target positions will be around the perimeter of the cards in the grid, avoiding separating the cards into two isolated groups. We have demonstrated that the goal hierarchy is the appropriate technique to guide the actions of the machine during the game.

Future work could be organized in two steps. First, we could try to implement a machine-machine version of the Yōkai game and compare the effectiveness of this collaboration with our results in the $\mathfrak{h}$-$\mathfrak{m}$ and $\mathfrak{h}$-$\mathfrak{h}$ configuration. Second, we could include a machine learning module in the system architecture, which will endow the machine agent with the necessary skills to learn new strategies based on previous games. Finally, we could detach our java interface and replace it with a web version that will allow us to make the game available to a wider audience.

References

[1] R. Fagin, J. Halpern, Y. Moses and M. Vardi, *Reasoning about Knowledge*, MIT Press, Cambridge, 1995.
[2] J.Y. Halpern and Y. Moses, A Guide to Completeness and Complexity for Modal Logics of Knowledge and Belief, *Artificial Intelligence* **54**(2) (1992), 319–379.
[3] J. Gerbrandy and W. Groeneveld, Reasoning about information change, *Journal of Logic, Language, and Information* **6** (1997), 147–196.
[4] J.A. Plaza, Logics of public communications, in: *Proceedings of the 4th International Symposium on Methodologies for Intelligent Systems*, M. Emrich, M. Pfeifer, M. Hadzikadic and Z. Ras, eds, 201-216, 1989.
[5] A. Baltag, L. Moss and S. Solecki, The Logic of Public Announcements, Common Knowledge and Private Suspicions, in: *Proceedings of the Seventh Conference on Theoretical Aspects of Rationality and Knowledge (TARK'98)*, I. Gilboa, ed., Morgan Kaufmann, San Francisco, CA, 1998, pp. 43–56.
[6] H.P. van Ditmarsch, W. van der Hoek and B. Kooi, *Dynamic Epistemic Logic*, Kluwer Academic Publishers, N.-Y., 2007. ISBN 1402058381.
[7] C. Lutz, Complexity and succinctness of public announcement logic, in: *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006)*, ACM, 2006, pp. 137–143.

[8] T. Bolander, H. van Ditmarsch, A. Herzig, E. Lorini, P. Pardo and F. Schwarzentruber, Announcements to Attentive Agents, *Journal of Logic, Language and Information* **25**(1) (2015), 1–35.
[9] G. Aucher and F. Schwarzentruber, On the complexity of dynamic epistemic logic, in: *Proceedings of the 14th Conference on Theoretical Aspects of Rationality and Knowledge (TARK 2013)*, 2013.
[10] T. Bolander and M.B. Andersen, Epistemic planning for single- and multi-agent systems, *Journal of Applied Non-Classical Logics* **21**(1) (2011), 656–680.
[11] E. Lorini, Rethinking epistemic logic with belief bases, *Artificial Intelligence* **282** (2020).
[12] E. Lorini, In Praise of Belief Bases: Doing Epistemic Logic Without Possible Worlds, in: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, AAAI Press, USA, 2018, pp. 1915–1922.
[13] Y. Shoham, Logical Theories of Intention and the Database Perspective, *Journal of Philosophical Logic* **38**(6) (2009), 633–648.
[14] E. Lorini and F. Romero, Decision procedures for epistemic logic exploiting belief bases, in: *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019)*, IFAAMAS, USA, 2019, pp. 944–952.
[15] T. Bolander, Seeing is believing: Formalising false-belief tasks in dynamic epistemic logic, in: *Proceedings of the European conference on Social Intelligence (ECSI-2014)*, A. Herzig and E. Lorini, eds, 2014, pp. 87–107.
[16] L. Dissing and T. Bolander, Implementing Theory of Mind on a Robot Using Dynamic Epistemic Logic, in: *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, (IJCAI 2020)*, C. Bessiere, ed., 2020, pp. 1615–1621.
[17] N. Bard, J.N. Foerster, S. Chandar, N. Burch, M. Lanctot, H.F. Song, E. Parisotto, V. Dumoulin, S. Moitra, E. Hughes, I. Dunning, S. Mourad, H. Larochelle, M.G. Bellemare and M. Bowling, The Hanabi challenge: A new frontier for AI research, *Artificial Intelligence* **280** (2020).
[18] M. Eger, C. Martens and M.A. Cordoba, An intentional AI for hanabi, in: *IEEE Conference on Computational Intelligence and Games (CIG 2017)*, IEEE, USA, 2017, pp. 68–75.
[19] M. Eger and C. Martens, Practical Specification of Belief Manipulation in Games, in: *Proceedings of the Thirteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-17)*, B. Magerko and J.P. Rowe, eds, AAAI Press, USA, 2017, pp. 30–36.
[20] G. Aucher, Private announcement and belief expansion: an internal perspective, *Journal of Logic and Computation* **22**(3) (2012), 451–479.
[21] T. Caridroit, J.-M. Lagniez, D. Le Berre, T. de Lima and V. Montmirail, A SAT-Based Approach for Solving the Modal Logic S5-Satisfiability Problem, in: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, AAAI Press, 2017, pp. 3864–3870.

Appendix A. Goals modeling

We introduce several additional definitions. First, we define the following abbreviation for $t \in TIME$ and $p \in GRID$:

$$\mathsf{emp}^t_p \quad \stackrel{\text{def}}{=} \quad \bigwedge_{x \in CARDS} \neg pos^t_{x,p}$$

emp^t_p reads "the position p is empty". Moreover:

$$OPOS : TIME \longrightarrow 2^{GRID}$$

$$t \mapsto OPOS^t = \{p \in GRID : \neg \mathsf{emp}^t_p \in \Sigma_f \text{ and } now^{\geqslant t} \in \Sigma_f\}$$

is a function returning the set of all occupied positions at time t.

A set of two different cards x and x', or a non-empty set X of cards, contains "neighboring cards" if these cards are located on neighboring positions ($X \subseteq CARDS : 3 \leqslant |X| \leqslant 4$) :

$$\mathsf{nbgCards}^t_{\{x,x'\}} \quad \stackrel{\text{def}}{=} \quad \bigvee_{\substack{p \in OPOS^t \\ p' \in NEIG(p)}} (pos^t_{x,p} \wedge pos^t_{x',p'})$$

$$\mathsf{nbgCards}^t_X \quad \stackrel{\text{def}}{=} \quad \bigvee_{\substack{x,x' \in X \\ x \neq x'}} \left(\mathsf{nbgCards}^t_{X \setminus \{x\}} \wedge \mathsf{nbgCards}^t_{\{x,x'\}}\right)$$

For every non empty $X, X' \subseteq CARDS : X \cap X' = \emptyset$, $\mathsf{hvECN}^t_{X,X'}$ is true iff there are a card in X and a card in X' that have at least one Empty Common Neighbor position. That is:

$$\mathsf{hvECN}^t_{X,X'} \quad \stackrel{\text{def}}{=} \quad \bigvee_{\substack{x \in X \\ p \in OPOS^t}} \bigvee_{\substack{x' \in X' \\ p' \in OPOS^t : \\ NEIG(p) \cap NEIG(p') \neq \emptyset}} (pos^t_{x,p} \wedge pos^t_{x',p'})$$

Finally, for every non empty $X \subset CARDS$, hvEN^t_X is true iff X has at leat one emtpy neighboor position :

$$\mathsf{hvEN}^t_X \quad \stackrel{\text{def}}{=} \quad \bigvee_{\substack{x \in X \\ p \in OPOS^t \\ p' \in NEIG(p)}} (pos^t_{x,p} \wedge \mathsf{emp}^t_{p'})$$

But if we consider that agent $\mathfrak{h}$ can make a mistake, then it is possible we need to remove such rules from the beliefs of the agent (when this rule leads to inconsistencies in $\mathfrak{m}$'s beliefs.

In the rest of this section, we associate to each action that the agent must perform a set of goals. It is the satisfaction of one of these goals that will cause the agent to perform a specific action. Among these goals, there is always one (called "default goal") corresponding to a strategy to be executed by default when all the other more specific "specific goals" have failed or have been judged unreachable. A goal is considered "attainable" when its precondition is satisfied before attempting to satisfy this goal. This precondition is there to inform the agent $\mathfrak{m}$ that it is not useful to look for a plan to satisfy this goal if this precondition is false. If the precondition is satisfied, this does not mean that the goal will always be satisfied, only that it is possible that it will be.

Of course, the set of specific goals given for each action is not complete, and one can imagine defining other specific goals. Likewise, there are probably other default goals. This is only to give examples and show how to proceed.

A.1. Observe actions

Here we describe a set of goals that will guide the agent $\mathfrak{m}$ to observe one card rather than another. If the agent fails to successfully execute a specific goal, then it will try to satisfy the default goal of looking at a random card.

(Specific) goal: observe a card around a cards group of the same color. This goal concerns the observation of a card around a set X of cards of the same color, known by the agent $\mathfrak{m}$. By definition, this means that $\mathfrak{m}$ aims there exists a card x' not belonging to X such that: the color of x' is not known by $\mathfrak{m}$ at time t, x' is in a position at least one card away from X at time t, and the next instant (at $t+1$) $\mathfrak{m}$ will know the color of x'.

A precondition that must be verified before trying to satisfy this goal and that all the cards of the set X are indeed cards located in neighboring positions at time t.

Let $X \subseteq CARDS : 1 \leqslant |X| \leqslant 3$, a set of cards of the same color:

$$\alpha^t_{obsArroundCards(X)} \stackrel{\text{def}}{=} \bigvee_{\substack{x' \in CARDS \\ x' \notin X}} \Big(\Big(\bigwedge_{c \in COLORS} \neg \triangle_{\mathfrak{m}}\, col^t_{x',c} \Big) \wedge \mathsf{nbgCards}^t_{X \cup \{x'\}} \wedge \Big(\bigvee_{c \in COLORS} \triangle_{\mathfrak{m}}\, col^{t+1}_{x',c} \Big) \Big)$$

$$\mathcal{P}_g(\alpha^t_{obsArroundCards(X)}) \stackrel{\text{def}}{=} \mathsf{nbgCards}^t_X$$

(Specific) goal: observe a card whose color is unknown. If agent $\mathfrak{m}$ considers that it is not possible to satisfy the previous goal, then $\mathfrak{m}$ will aim to observe the color of a card that it does not already know (this is the precondition of the action).

Let $x \in CARDS$:

$$\alpha^t_{obsUnknownColorCard(x)} \stackrel{\text{def}}{=} \bigvee_{c \in COLORS} \triangle_{\mathfrak{m}}\, col^{t+1}_{x,c}$$

$$\mathcal{P}_g(\alpha^t_{obsUnknownColorCard(x)}) \stackrel{\text{def}}{=} \bigwedge_{c \in COLORS} \neg \triangle_{\mathfrak{m}}\, col^t_{x,c}$$

(default) goal: observe a card randomly. When all other goals failed or were impossible to achieve, then agent $\mathfrak{m}$ performs the default goal of observing a random card. Because of the previous goal, $\mathfrak{m}$ already knows the color of this card, but it is still obliged by the rules of the game to observe the color of a card. This action is always executable (no precondition).

Let $x \in CARDS$:

$$\alpha^t_{obsRandomly(x)} \stackrel{\text{def}}{=} \bigvee_{c \in COLORS} \triangle_{\mathfrak{m}}\, col^{t+1}_{x,c}$$

$$\mathcal{P}_g(\alpha^t_{obsRandomly(x)}) \stackrel{\text{def}}{=} \top$$

How to compute parameters of goals?. The parameters of a goal are the free variables in the definition of this goal. For convenience, we define a useful function in Algorithm 1.

The GETSUBSETSGTLEQ function returns all the subsets of the set X whose number of elements is strictly greater than τ_m and is lower or equal to τ_M. For example, GETSUBSETSGTLEQ$(\{1,2\}, 0, |\{1,2\}|)$ returns $\{\{1\},\{2\},\{1,2\}\}$ (the empty set is not in the answer but $\{1,2\}$ is). Moreover, the GETELEMENTSOFSIZE function returns all elements of the input parameter X whose length is exactly ω (X is a set of sets, and $\omega \in \mathbb{N}$).

Goals parameters, the ordering of the goals and the verification of their precondition are determined by the meta-logical Algorithm 2.

Algorithm 1 Userful functions for goals parameters computation

function GETSUBSETSGTLEQ(X, τ_m, τ_M)
  $S \leftarrow \{\}$
  for all $X' \in 2^X$ **do**
    if $\tau_m < |X'| \leqslant \tau_M$ **then**
      PUSH(X', S)
    end if
  end for
  return S
end function

function GETELEMENTSOFSIZE(X, ω)
  $S \leftarrow \{\}$
  for all $X' \in X$ **do**
    if $|X'| == \omega$ **then**
      PUSH(X', S)
    end if
  end for
  return S
end function

Algorithm 2 Computation of the arguments for observe goals?

function TRYTOOBSERVE(S, α^t_g)
  success $\leftarrow$ false
  while $S \neq \emptyset$ && ! *success* **do**
    $x \leftarrow$ POP(S)
    if $\mathcal{P}_g(\alpha^t_{g(x)})$ **then**
      success $\leftarrow$ CHECKPLAN($\alpha^t_{g(x)}$)
    end if
  end while
  return *success*
end function

$S \leftarrow \{\}$
for all $c \in COLORS$ **do**
  $X_c \leftarrow \{x \in CARDS : \triangle_{\mathrm{m}}\, col^t_{x,c} \in \Sigma_f\}$
  $S \leftarrow S \cup$ GETSUBSETSGTLEQ($X_c, 0, |X_c|$)
end for
$S' \leftarrow$ SORTBYDECREASINGSIZE(S) ▷ $\forall X_i, X_{i+1} \in S', |X_i| \geqslant |X_{i+1}|$

if ! TRYTOOBSERVE($S', \alpha^t_{obsArroundCards}$) **then**
  $X \leftarrow \{x \in CARDS : \exists c \in COLORS, \triangle_{\mathrm{m}}\, col^t_{x,c} \in \Sigma_f\}$
  $X' \leftarrow CARDS \setminus X$ ▷ The set of cards whose color is unknown
  if ! TRYTOOBSERVE($X', \alpha^t_{obsUnknownColorCard}$) **then**
    TRYTOOBSERVE($CARDS, \alpha^t_{obsRandomly}$)
  end if
end if

In this algorithm, the TRYTOOBSERVE function checks, for each element x of S (line 2.4), that the goal precondition for x is true (line 2.5), and if so attempts to accomplish the goal for x (line 2.6). If at least one attempt is successful the function returns true. If for every x the precondition is false or the goal not satisfied, the function returns false. Note that the function parameter S is a set of elements that can be cards or non empty subsets of cards.

So, in Algorithm 2, we start by computing the set S which contains, for each color (line 2.13), the set of cards of this color that are known by the agent $\mathfrak{m}$ (line 2.14), as well as all its non-empty subsets (line 2.15). Finally, S' (line 2.17) is the set such that: each element is a subset of cards of the same color, and all these elements are sorted by size decreasing. For example, suppose that:

$$\Sigma_f = \{\triangle_{\mathfrak{m}}\, col^t_{15,b}, \triangle_{\mathfrak{m}}\, col^t_{6,b}, \triangle_{\mathfrak{m}}\, col^t_{1,g}, \triangle_{\mathfrak{m}}\, col^t_{13,g}, \triangle_{\mathfrak{m}}\, col^t_{7,g}, \triangle_{\mathfrak{m}}\, col^t_{9,y}\}.$$

So, we have:

$$S = \{\{15,6\},\{15\},\{6\},\{1,13,7\},\{1,13\},\{1,7\},\{13,7\},\{1\},\{13\},\{7\},\{9\}\}$$
$$S' = \{\{1,13,7\},\{15,6\},\{1,13\},\{1,7\},\{13,7\},\{15\},\{6\},\{1\},\{13\},\{7\},\{9\}\}$$

The end of Algorithm 2 describes the ordering of the different goals: the agent $\mathfrak{m}$ will first try to observe a card around a group of neighboring cards of the same color (line 2.19), then it will try to look at a random card among those it does not know (line 2.20 to line 2.22), then if no previous goal has been achieved, then he will look at a random card (line 2.23).

A.2. Move actions

Goal: group 4 cards of the same color. Let $X \subseteq CARDS$ such that $|X| = 4$ and every cards in X have the same color:

$$\alpha^t_{groupCards_4(X)} \stackrel{\text{def}}{=} \mathsf{nbgCards}^{t+1}_X$$

$$\mathcal{P}_g(\alpha^t_{groupCards_4(X)}) \stackrel{\text{def}}{=} \neg\mathsf{nbgCards}^t_X \wedge \left(\bigvee_{\substack{X' \subseteq X \\ |X'|=3}} (\mathsf{nbgCards}^t_{X'} \wedge \mathsf{hvEN}^t_{X'}) \vee \bigvee_{\substack{X' \subseteq X \\ |X'|=2}} (\mathsf{nbgCards}^t_{X'} \wedge \mathsf{hvECN}^t_{X \setminus X', X'}) \right)$$

The precondition above reads that there exists a plan to group the 4 cards in X iff: 1) they are currently not neighboring cards; and 2) either a subset X' of 3 cards is a set of neighboring cards and at least one card of X' has a free neighboring position, or a subset X' of 2 cards is a set of neighboring cards and at least one of the other cards of X (not in X') has a neighboring free position in common with a card of X'.

Goal: group 3 cards of the same color. Let $c \in COLORS$, $X \subseteq CARDS$ such that $|X| = 3$ and every cards in X have the same color:

$$\alpha^t_{groupCards_3(X)} \stackrel{\text{def}}{=} \mathsf{nbgCards}^{t+1}_X$$

$$\mathcal{P}_g(\alpha^t_{groupCards_3(X)}, c) \stackrel{\text{def}}{=} \neg\mathsf{nbgCards}^t_X \wedge \bigwedge_{x \in CARDS \setminus X} \left(col^t_{x,c} \rightarrow \neg\mathsf{nbgCards}^t_{X \cup \{x\}} \right) \wedge \left(\bigvee_{\substack{X' \subseteq X \\ |X'|=2}} (\mathsf{nbgCards}^t_{X'} \wedge \mathsf{hvEN}^t_{X'}) \vee \bigvee_{\substack{x, x' \in X \\ x \neq x'}} \mathsf{hvECN}^t_{\{x\},\{x'\}} \right)$$

The precondition above reads that there exists a plan to group the 3 cards in X having the same color c iff: 1) the cards in X are currently not neighboring cards; 2) there does not exist a card of color c not in X forming (together with X) a set of neighboring cards ; and 3) either a subset X' of 2 cards is a set of neighboring cards and at least one card of X' has a free neighboring position, or at least 2 cards in X have a neighboring free position in common.

Goal: group 2 cards of the same color. Let $c \in COLORS$, $X \subseteq CARDS$ such that $|X| = 2$ and every cards in X have the same color:

$$\alpha^t_{groupCards_2(X)} \stackrel{\text{def}}{=} \mathsf{nbgCards}^{t+1}_X$$

$$\mathcal{P}_g(\alpha^t_{groupCards_2(X)}, c) \stackrel{\text{def}}{=} \neg\mathsf{nbgCards}^t_X \wedge \bigwedge_{x \in CARDS \setminus X} \Big(col^t_{x,\mathrm{g}} \rightarrow \bigwedge_{x' \in X} \neg\mathsf{nbgCards}^t_{\{x,x'\}} \Big) \wedge \bigvee_{x \in X} \mathsf{hvEN}^t_{\{x\}}$$

The precondition above reads that there exists a plan to group 2 cards x and x' having the same color c iff: 1) the cards in X are currently not neighboring cards; 2) there does not exist a card of color c not in X forming, together with a card in X, a set of neighboring cards ; and 3) at least 1 card in X has a neighboring free position in common.

Goal: move randomly a card. Let $x \in CARDS$:

$$\alpha^t_{randomMove(x)} \stackrel{\text{def}}{=} \bigvee_{\substack{p \in OPOS^t \\ p' \in GRID \setminus OPOS^t}} (pos^t_{x,p} \wedge pos^{t+1}_{x,p'})$$

$$\mathcal{P}_g(\alpha^t_{randomMove(x)}) \stackrel{\text{def}}{=} \Big(\bigwedge_{c \in COLORS} \neg \triangle_{\mathfrak{m}}\, col^t_{x,c} \Big) \vee \bigvee_{c \in COLORS} \Big(col^t_{x,c} \wedge \bigwedge_{\substack{x' \in CARDS \\ x' \neq x}} (col^t_{x',c} \rightarrow \neg\mathsf{nbgCards}^t_{\{x,x'\}}) \Big)$$

How to compute arguments of goals?. The parameters of goals and their precondition are computed directly from the mutable belief base Σ_m of agent $\mathfrak{m}$ in the following way (see Algorithm 3).

In Algorithm 3, the function GETMARKEDCARDS has a parameter X (a set of cards having a same color). If X does not already contain 4 cards (line 3.4), then this function will return the set of all cards x such that: 1) there exist a hint h that includes the color c of cards in X (line 3.5); 2) x is marked by h (line 3.6) and 3) $\mathfrak{m}$ believes that x does not have a color not represented in h (that is, $\mathfrak{m}$ has no reason to believe that h gives a false indication about the color of x). Note that H_c is computed only if the number of known cards is lower than 4 (line 3.4) because when all the cards of a same color are known, we do not try to generate moves near cards with a hint.

The function TRYTOMOVE has three parameters: a set S of elements (either subsets of cards, or cards), a given goal α^t_g and a boolean b. The only difference of this function from the TRYTOOBSERVE function is the case where b is true (line 3.16): in this case, an additional parameter (the color of cards in X_c) is used to check the goal precondition. This additional parameter is required when α^t_g is either $\alpha^t_{groupCards_3}$ (to group 3 cards) or $\alpha^t_{groupCards_2}$ (to group 2 cards), which indicates a true value for b.

Finally, for each color c (line 3.29) we compute: X_c (line 3.30) that is the set of cards of color c which are known by $\mathfrak{m}$; H_c (line 3.31) that is the set of cards unknown by agent $\mathfrak{m}$ but marked by a hint containing color c. Then, we generate the set of subsets of cards (line 3.32) for which either the color is c or the color is unknown but these cards are marked with a hint containing color c, and we only retain the subsets having between 2 and 4 cards. As it is preferable to first try to group together the greatest possible number of cards of the same color, and preferably cards of which we know the color (that is, not marked by a hint), we sort all the subsets of cards using the **sort** function (line 3.34). So, SORTBYDECREASINGSIZEHINTS$(S) = \{X_1, X_2, ..., X_n\}$ is a set of subsets of cards such that, for every $i \in [1..n-1]$: 1) $|X_i| \geqslant |X_{i+1}|$; 2) if $|X_i| = |X_{i+1}|$ then $|\mathbf{hints}^t(X_i)| \geqslant |\mathbf{hints}^t(X_{i+1})|$, where $\mathbf{hints}^t(X) = \{x \in X : \exists h \in HINTS, mark^t_{x,h} \in \Sigma_f\}$ is the set of cards in X marked by a hint.

Algorithm 3 Computation of the arguments for move goals?

function GETMARKEDCARDS(X)
  ▷ X is a set of cards of the same color
  $c \leftarrow$ GETCOLOR(X) ▷ return the color of cards in X
  if $|X| < 4$ **then**
    $H_c \leftarrow \{x \in CARDS : \exists h \in HINTS$ such that $c \in h$ and
      $\Sigma_m \cup \Sigma_c \models \Box_{\mathfrak{m}} \left(mark^t_{x,h} \wedge \bigwedge_{c' \in COLORS \setminus h} \neg col^t_{x,c'}\right)\}$
  else
    $H_c \leftarrow \emptyset$
  end if
end function

function TRYTOMOVE(S, α^t_g, b)
  $success \leftarrow$ false
  while $S \neq \emptyset$ && ! $success$ **do**
    $x \leftarrow$ POP(S)
    if b **then**
      $Precond \leftarrow \mathcal{P}_g(\alpha^t_{g(x)}$, GETCOLOR($x$))
    else
      $Precond \leftarrow \mathcal{P}_g(\alpha^t_{g(x)})$
    end if
    if b **then**
      $success \leftarrow$ CHECKPLAN($\alpha^t_{g(x)}$)
    end if
  end while
  return $success$
end function

$S \leftarrow \{\,\}$
for $c \in COLORS$ **do**
  $X_c \leftarrow \{x \in CARDS : \triangle_{\mathfrak{m}}\, col^t_{x,c} \in \Sigma_m\}$
  $H_c \leftarrow$ GETMARKEDCARDS(X_c)
  $S \leftarrow$ GETSUBSETSGTLEQ($X_c \cup H_c, 1, 4$)
end for
$S' \leftarrow$ SORTBYDECREASINGSIZEHINTS(S) ▷ See below for explanations

$X_4 \leftarrow$ GETELEMENTSOFSIZE($S', 4$)
if ! TRYTOMOVE($X_4, \alpha^t_{groupCards_4}$, false) **then**
  $X_3 \leftarrow$ GETELEMENTSOFSIZE($S', 3$)
  if ! TRYTOMOVE($X_3, \alpha^t_{groupCards_3}$, true) **then**
    $X_2 \leftarrow$ GETELEMENTSOFSIZE($S', 2$)
    if ! TRYTOMOVE($X_2, \alpha^t_{groupCards_2}$, true) **then**
      TRYTOMOVE($CARDS, \alpha^t_{randomMove}$, false)
    end if
  end if
end if

In the rest of Algorithm 3, for each subset of S' we try to group sets of cards of decreasing size (line 3.36 to line 3.41). If no plan is successful, then the default goal (move a card randomly, see line 3.42) is executed (which is always executable).

A.3. Mark actions and active actions

Goal: mark a card unknown by $\mathfrak{h}$ *with the smallest hint available.* Let $x \in CARDS$ be a cards that is not marked at time t and $h \in HINTS$ a hint that is active at time t:

$$\alpha^t_{markUnknownCard(x,h)} \stackrel{\text{def}}{=} mark^{t+1}_{x,h}$$

$$\mathcal{P}_g(\alpha^t_{markUnknownCard(x,h)}) \stackrel{\text{def}}{=} \bigwedge_{c \in COLORS} \neg \triangle^t_{\mathfrak{h}}\, col^t_{x,c} \wedge \bigvee_{c' \in h} \triangle_{\mathfrak{m}}\, col^t_{x,c'}$$

Agent $\mathfrak{m}$ has a goal to mark with a hint h a card x for which the color is unknown by agent $\mathfrak{h}$ at time t if and only if x is marked by h at time $t+1$. Note it is not necessary to check if x is not marked at time t because we already suppose it is the case for the x goal parameter.

The precondition ensures that $\mathfrak{h}$ does not know the color of card x yet at time t, and that there exists a color c' in the hint h such that $\mathfrak{m}$ believes that x has the color c'.

Goal: mark randomly a card. Let $x \in CARDS$ be a cards that is not marked at time t and $h \in HINTS$ a hint that is active at time t:

$$\alpha^t_{markRandomCard(x,h)} \stackrel{\text{def}}{=} mark^{t+1}_{x,h}$$

$$\mathcal{P}_g(\alpha^t_{markRandomCard(x,h)}) \stackrel{\text{def}}{=} \bigvee_{c' \in h} \triangle_{\mathfrak{m}}\, col^t_{x,c'}$$

The goal is less restrictive than the previous one, since the fact that agent $\mathfrak{h}$ does not know the color of the card that agent $\mathfrak{m}$ is trying to mark is not a precondition for trying to reach this goal. Thus, this goal allows to play in the case where at least one hint is active but its use will not be optimally informative for $\mathfrak{h}$ (who already knows the color of the marked card).

The precondition just ensures there exists a color c' in the hint h such that $\mathfrak{m}$ believes that x has the color c'.

How to compute arguments of goals?. The parameters of goals and their precondition are computed directly from the mutable beliefs base Σ_m of agent $\mathfrak{m}$ following Algorithm 4.

In Algorithm 4, we first define the TRYTOMARK function which is similar to TRYTOMOVE and TRYTOOBSERVE previous functions. This function try, for each hint in parameter H (line 4.3) and each card in parameter X (line 4.6), to check precondition of the goal in parameter α^t_g (line 4.8). If this precondition is satisfied, the algorithm try to satisfy the goal itself (line 4.9). As soon as a goal is satisfied (a plan has been found) the true value is returned, else false is returned.

So, x contains the set of cards that are not marked at time t (line 4.16), and H contains the set of active hints at time t (line 4.17).

So, we compute H_I (line 4.18) that gets elements of H sorted following increasing size (that is, for every $h_j, h_{j+1} \in H_I, |h_j| \leqslant |h_{j+1}|$), and H_D (line 4.20) that gets elements of H sorted following decreasing size (that is, for every $h_j, h_{j+1} \in H_D, |h_j| \geqslant |h_{j+1}|$).

In the rest of Algorithm 4, we try first to satisfy the goal "to mark an unknown card x with an active hint h". If it is not possible to satisfy this goal or its preconditions, or if there is no active clue, or if there is no card whose color is unknown to the agent $\mathfrak{h}$ (line 4.19), then the agent $\mathfrak{m}$ will try to satisfy the next goal "to mark a randomly chosen card x with an active hint h" (line 4.21).

Note that when $\mathfrak{m}$ seeks to satisfy the first goal, the set of available indices are classified according to increasing order with respect to their size (i.e., with respect to the number of colors that defines them). This is because the smaller the size of a hint, the more informative that hint is. (A hint of size 1 actually tells agent $\mathfrak{h}$ the real color

Algorithm 4 Computation of the arguments for mark goals?

function TRYTOMARK(X, H, α^t_g)
  $success \leftarrow$ false
  while $H \neq \emptyset$ && ! $success$ **do**
    $h \leftarrow$ POP(H)
    $S_X \leftarrow X$
    while $S_X \neq \emptyset$ && ! $success$ **do**
      $x \leftarrow$ POP(S_X)
      if $\mathcal{P}_g(\alpha^t_{g(x,h)})$ **then**
        $success \leftarrow$ CHECKPLAN($\alpha^t_{g(x,h)}$)
      end if
    end while
  end while
  return $success$
end function

$X \leftarrow \left\{x \in CARDS : \Sigma_m \cup \Sigma_c \models \bigvee_{t \in TIME} \left(now^{=t} \wedge \Box_{\mathfrak{m}} \bigwedge_{h \in HINTS} \neg mark^t_{x,h}\right)\right\}$
$H \leftarrow \left\{h \in HINTS : \Sigma_m \cup \Sigma_c \models \bigvee_{t \in TIME} \left(now^{=t} \wedge \Box_{\mathfrak{m}}\, active^t_h\right)\right\}$
$H_I \leftarrow$ SORTBYINCREASINGSIZE(H)
if ! TRYTOMARK($X, H_I, \alpha^t_{markUnknownCard}$) **then**
  $H_D \leftarrow$ SORTBYDECREASINGSIZE(H)
  if ! TRYTOMARK($X, H_D, \alpha^t_{markRandomCard}$) **then**
    $h \leftarrow$ RANDOM($HINTS \setminus H$) ▷ a non active hint
    ACTIVE(h)
  end if
end if

of the card.) Conversely, when $\mathfrak{m}$ tries to satisfy the second goal, we use the list of hints classified according to their decreasing size. In other words, we start by trying to satisfy the goal with the largest hint (therefore, the least informative hint for agent $\mathfrak{h}$). This is simply because, in this case, there is no card whose color $\mathfrak{h}$ does not know, so we try to use the least informative hint first (in order to keep the most informative hints for the end of the game).

If no goal is satisfiable, then the agent $\mathfrak{m}$ activates a hint randomly chosen among the set of indices which are not yet active.

Note that in the game, the following situation is possible: agent $\mathfrak{m}$ does not know any green card, but the only remaining hint has already been enabled and its color is blue. In this case, the previous algorithm allow $\mathfrak{m}$ to not execute any action related to hint ($\mathfrak{m}$ passes its turn).

Appendix B. Sketch of proof of Theorem 1

Hardness is clear since $\mathcal{L}(ATM)$ extends the propositional logic language. As for membership, we can find a polysize satisfiability preserving translation from $\mathcal{L}(ATM)$ to propositional logic. The translation is divided in two steps. First, we translate the input formula in $\mathcal{L}(ATM)$ into a formula of a restricted mono-modal language with no nesting of the modal operator. Secondly, we translate the latter into a propositional logic language in a way similar to the standard translation of modal logic into FOL. We take care of translating a finite theory including axioms corresponding to the three constraints of Definition 1. The theory is finite since we only need to consider instances of the axioms whose symbols occur in the input formula.

In the rest of this section, we detail the steps of the proof.

Step 1. We define the following modal language $\mathcal{L}^{Mod}$ into which the language $\mathcal{L}$ will be translated:

$$\begin{aligned} \omega &::= q \mid \neg\omega \mid \omega_1 \wedge \omega_2, \\ \varphi &::= q \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \blacksquare\,\omega \end{aligned}$$

where q ranges over the following set of atomic formulas:

$$AFML = \{atm_{\triangle_{\mathrm{m}}\,\alpha} : \alpha \in \mathcal{L}_0\} \cup \{atm_x : x \in \mathcal{L}_0^T(ATM)\}.$$

We interpret the language $\mathcal{L}^{Mod}$ w.r.t. a pair (M, w), called pointed Kripke model, where $M = (W, \Rightarrow, \pi)$, W is a set of worlds, $\Rightarrow \subseteq W \times W$ and $\pi : AFML \longrightarrow 2^W$. (Boolean cases are again omitted as they are defined in the usual way.)

Definition 8. *The semantic interpretation for formulas in $\mathcal{L}^{Mod}$ w.r.t. a pointed Kripke model (M, w) is as follows:*

$$\begin{aligned} (M, w) &\models q \Longleftrightarrow w \in \pi(q); \\ (M, w) &\models \blacksquare\,\omega \Longleftrightarrow \forall v \in W, \text{ if } w \Rightarrow v \text{ then } (M, v) \models \omega. \end{aligned}$$

The class of pointed Kripke models is noted **K**. Satisfiability and validity of formulas in $\mathcal{L}^{Mod}$ relative to the class **K** is defined in the usual way.

Let $tr_1 : \mathcal{L} \longrightarrow \mathcal{L}^{Mod}$ be the following translation:

$$\begin{aligned} tr_1(x) &= atm_x \text{ for } x \in \mathcal{L}_0^T, \\ tr_1(\neg\varphi) &= \neg tr_1(\varphi), \\ tr_1(\varphi_1 \wedge \varphi_2) &= tr_1(\varphi_1) \wedge tr_1(\varphi_2), \\ tr_1(\triangle_{\mathrm{m}}\,\alpha) &= atm_{\triangle_{\mathrm{m}}\,\alpha} \wedge \blacksquare\, tr_0(\alpha), \\ tr_1(\Box_{\mathrm{m}}\,\beta) &= \blacksquare\, tr_0(\beta), \end{aligned}$$

with $tr_0 : \mathcal{L}_0 \longrightarrow \mathcal{L}^{Mod}$ such that:

$$\begin{aligned} tr_0(x) &= atm_x \text{ for } x \in \mathcal{L}_0^T, \\ tr_0(\triangle_{\mathrm{m}}\,\alpha) &= atm_{\triangle_{\mathrm{m}}\,\alpha}, \\ tr_0(\neg\alpha) &= \neg tr_0(\alpha), \\ tr_0(\alpha_1 \wedge \alpha_2) &= tr_0(\alpha_1) \wedge tr_0(\alpha_2). \end{aligned}$$

As the following theorem indicates, the polynomial translation tr_1 guarantees the transfer of satisfiability from model class **M** to model class **K**.

Proposition 2. *Let $\varphi \in \mathcal{L}$. Then, φ is satisfiable in the class* **M** *if and only if $\blacksquare\left(\bigwedge_{\omega\in\Gamma_\varphi}\omega\right) \wedge tr_1(\varphi)$ is satisfiable in the class* **K**, *where Γ_φ is defined as follows:*

$$\begin{aligned} \Gamma_\varphi = &\{atm_{now^{\geqslant 0}}\} \cup \{atm_{now^{\geqslant t}} \rightarrow atm_{now^{\geqslant t'}} : t' \leqslant t \text{ and } now^{\geqslant t}, now^{\geqslant t'} \in SF(\varphi)\} \cup \\ &\{atm_{now^{\geqslant t}} \leftrightarrow atm_{\triangle_{\mathrm{m}}\, now^{\geqslant t}} : now^{\geqslant t} \in SF(\varphi)\}, \end{aligned}$$

and $SF(\varphi)$ is the set of subformulas of φ which is inductively defined as follows:

$$
\begin{aligned}
&SF(p^t) = \{p^t\},\\
&SF\left(now^{\geqslant t}\right) = \{now^{\geqslant t}\},\\
&SF\left(\triangle^t_{\mathfrak{h}}\,\alpha\right) = \{\triangle^t_{\mathfrak{h}}\,\alpha\} \cup SF(\alpha),\\
&SF(\triangle_{\mathfrak{m}}\,\alpha) = \{\triangle_{\mathfrak{m}}\,\alpha\} \cup SF(\alpha),\\
&SF(\neg\varphi) = \{\neg\varphi\} \cup SF(\varphi),\\
&SF(\varphi_1 \wedge \varphi_2) = \{\varphi_1 \wedge \varphi_2\} \cup SF(\varphi_1) \cup SF(\varphi_2),\\
&SF(\Box_{\mathfrak{m}}\,\beta) = \{\Box_{\mathfrak{m}}\,\beta\} \cup SF(\beta).
\end{aligned}
$$

Step 2. As a second step, we provide a polysize reduction of $\mathcal{L}^{Mod}$-satisfiability to SAT, where the underlying propositional logic language $\mathcal{L}^{PL}$ is built from the following set of atomic propositions:

$$AFML^{+} = \{q_x : q \in AFML \text{ and } x \in \mathbb{N}\} \cup \{r_{x,y} : x, y \in \mathbb{N}\}.$$

Let $tr_2 : \mathcal{L}^{Mod} \times \mathbb{N} \times \mathbb{N} \longrightarrow \mathcal{L}^{PL}$ be the following translation function:

$$
\begin{aligned}
tr_2(q, x, y) &= q_x,\\
tr_2(\neg\varphi, x, y) &= \neg tr_2(\varphi, x, y),\\
tr_2(\varphi_1 \wedge \varphi_2, x, y) &= tr_2(\varphi_1, x, y) \wedge tr_2(\varphi_2, x, y),\\
tr_2(\blacksquare\,\omega, x, y) &= \bigwedge_{0 \leqslant z \leqslant y} \big(r_{x,z} \rightarrow tr_2(\omega, z, y)\big).
\end{aligned}
$$

Translation tr_2 is similar to the translation of modal logic S5 into propositional logic given in [21] and, more generally, to the standard translation of modal logic into FOL in which accessibility relations are encoded by special predicates.

The size of an $\mathcal{L}^{Mod}$ formula, $size(\varphi)$, is defined by:

$$
\begin{aligned}
size(q) &= 1,\\
size(\varphi_1 \wedge \varphi_2) &= size(\varphi_1) + size(\varphi_2) + 1,\\
size(\neg\varphi) &= size(\varphi) + 1,\\
size(\blacksquare\,\omega) &= size(\omega) + 1.
\end{aligned}
$$

Note that the size of $tr_2\big(\varphi, 0, size(\varphi)\big)$ is polynomial in the size of φ.

Proposition 3. *Let $\varphi \in \mathcal{L}^{Mod}$. Then, φ is satisfiable in the class* **K** *if and only if $tr_2\big(\varphi, 0, size(\varphi)\big)$ is satisfiable in propositional logic.*

Note that the size of $tr_2\big(\varphi, 0, size(\varphi)\big)$ is polynomial in the size of φ. Therefore, Theorem 1 follows from Proposition 2 and Proposition 3.

Appendix C. Sketch of proof of Theorem 2

The following equivalences are valid in the class **M**:

$$[+^t_{\mathfrak{m}}\alpha]\alpha' \leftrightarrow \begin{cases} \top, & \text{if } \alpha' = \triangle_{\mathfrak{m}}\,\alpha \text{ or } (\alpha' = now^{\geqslant t'} \text{ and } t' \leqslant t) \text{ or} \\ & (\alpha' = \triangle_{\mathfrak{m}}\, now^{\geqslant t'} \text{ and } t' \leqslant t), \\ \alpha', & \text{otherwise;} \end{cases}$$

$$[+^t_{\mathfrak{m}}\alpha]\neg\varphi \leftrightarrow \neg[+^t_{\mathfrak{m}}\alpha]\varphi;$$

$$[+^t_{\mathfrak{m}}\alpha](\varphi_1 \wedge \varphi_2) \leftrightarrow [+^t_{\mathfrak{m}}\alpha]\varphi_1 \wedge [+^t_{\mathfrak{m}}\alpha]\varphi_2;$$

$$[+^t_{\mathfrak{m}}\alpha]\,\Box_{\mathfrak{m}}\,\alpha' \leftrightarrow \Box_{\mathfrak{m}}\,\big((\alpha \wedge \bigwedge_{t' \leqslant t} now^{\geqslant t'}) \rightarrow \alpha'\big).$$

Thanks to these equivalences we can define the following reduction *red* transforming every $\mathcal{L}^+$ formula φ into an equivalent $\mathcal{L}$ formula $red(\varphi)$:

$$\begin{aligned}
red(p^t) &= p^t, \\
red(\triangle^t_{\mathfrak{h}}\,\alpha) &= \triangle^t_{\mathfrak{h}}\,\alpha, \\
red(\triangle_{\mathfrak{m}}\,\alpha) &= \triangle_{\mathfrak{m}}\,\alpha, \\
red(\neg\varphi) &= \neg red(\varphi), \\
red(\varphi_1 \wedge \varphi_2) &= red(\varphi_1) \wedge red(\varphi_2), \\
red(\Box_{\mathfrak{m}}\,\varphi) &= \Box_{\mathfrak{m}}\, red(\varphi), \\
red([+^t_{\mathfrak{m}}\alpha]\alpha') &= \begin{cases} \top, & \text{if } \alpha' = \triangle_{\mathfrak{m}}\,\alpha \text{ or } (\alpha' = now^{\geqslant t'} \text{ and } t' \leqslant t) \text{ or} \\ & (\alpha' = \triangle_{\mathfrak{m}}\, now^{\geqslant t'} \text{ and } t' \leqslant t), \\ red(\alpha'), & \text{otherwise;} \end{cases} \\
red([+^t_{\mathfrak{m}}\alpha]\neg\varphi) &= red(\neg[+^t_{\mathfrak{m}}\alpha]\varphi), \\
red\big([+^t_{\mathfrak{m}}\alpha](\varphi_1 \wedge \varphi_2)\big) &= red([+^t_{\mathfrak{m}}\alpha]\varphi_1 \wedge [+^t_{\mathfrak{m}}\alpha]\varphi_2), \\
red([+^t_{\mathfrak{m}}\alpha]\,\Box_{\mathfrak{m}}\,\alpha') &= red\Big(\,\Box_{\mathfrak{m}}\,\big((\alpha \wedge \bigwedge_{t' \leqslant t} now^{\geqslant t'}) \rightarrow \alpha'\big)\Big).
\end{aligned}$$

Proposition 4. *Let $\varphi \in \mathcal{L}^+$. Then, $\varphi \leftrightarrow red(\varphi)$ is valid in the class* **M***, and $red(\varphi) \in \mathcal{L}$.*

Theorem 2 is a consequence of Theorem 1, Proposition 4 and the fact that the size of $red(\varphi)$ is polynomial in the size of φ.