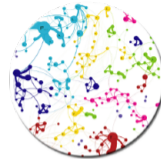




CNRS - INP - UT3 - UT1 - UT2J

Institut de Recherche en Informatique de Toulouse



# An introduction to cognitive planning

J. Fernandez, D. Longin, E. Lorini, F. Maris  
IRIT, Université de Toulouse, CNRS, Toulouse INP, UT3  
{prenom.nom}@irit.fr



# Plan of the tutorial

- 1 Logical framework for cognitive planning (Emiliano)
- 2 **QBF encoding of cognitive planning and TouIST platform (Frédéric)**  
**NOW!**
  - $\exists\forall$  optimal encoding of simple cognitive planning
  - TouIST: Toulouse Integrated Satisfiability Tool
- 3 Implementation and applications (Jorge)
- 4 Formal extension and implementation for collaborative gaming (Dominique & Jorge)



# Part 2 - QBF encoding of cognitive planning and TouIST platform

≡ optimal encoding



## ES-INF planning problem (recall)

A **solution plan** to an ES-INF planning problem  $\langle \Sigma, Op, \alpha_G \rangle$  is a sequence of operators  $\epsilon_1, \dots, \epsilon_k$  from  $Op$  such that  $\Sigma \models_{\mathbf{M}} \langle\langle \epsilon_1 \rangle\rangle \dots \langle\langle \epsilon_k \rangle\rangle \Box_m \alpha_G$

### Theorem

*An ES-INF planning problem  $\langle \Sigma, Op, \alpha_G \rangle$  has a solution plan if and only if it has a poly-size solution plan  $\epsilon_1, \dots, \epsilon_k$  with  $k \leq |Op|$  and  $\epsilon_i \neq \epsilon_j$  for all  $i < j$ .*

### Theorem

*Checking plan existence for an ES-INF planning problem is  $\Sigma_2^P$ -complete.*



# $\exists\forall$ encoding of ES-INF planning problem $\langle \Sigma, Op, \alpha_G \rangle$

## Theorem

Let  $\varphi \in \mathcal{L}_{\text{Frag}}^+$  and let  $\Sigma \subset \mathcal{L}_0$  be finite.

Then,  $\Sigma \models_{\text{M}} \varphi$  if and only if  $\models_{\text{M}} \bigwedge_{\alpha \in \Sigma} \Box_{\text{m}} \alpha \rightarrow \varphi$ .

A plan candidate  $\epsilon_1, \dots, \epsilon_k$  is a solution plan for  $\langle \Sigma, Op, \alpha_G \rangle$  if and only if the following formula is valid:

$$\left( \bigwedge_{\alpha \in \Sigma} \Box_{\text{m}} \alpha \right) \rightarrow \langle\langle \epsilon_1 \rangle\rangle \dots \langle\langle \epsilon_{k-1} \rangle\rangle \langle\langle \epsilon_k \rangle\rangle \Box_{\text{m}} \alpha_G$$

which can be also written as:

$$\left( \bigwedge_{\alpha \in \Sigma} \Box_{\text{m}} \alpha \right) \rightarrow \left( \bigwedge_{i \in \{1, \dots, k\}} [+_{\text{m}} \alpha_{\epsilon_1}] \dots [+_{\text{m}} \alpha_{\epsilon_{i-1}}] \mathcal{P}(\epsilon_i) \right) \wedge [+_{\text{m}} \alpha_{\epsilon_1}] \dots [+_{\text{m}} \alpha_{\epsilon_{k-1}}] [+_{\text{m}} \alpha_{\epsilon_k}] \Box_{\text{m}} \alpha_G$$



# $\exists\forall$ encoding of ES-INF planning problem $\langle \Sigma, Op, \alpha_G \rangle$

- Global selector variables  $V_s = \{s_{\epsilon \preceq \epsilon'} : \epsilon, \epsilon' \in Op\} \subseteq Atm$
- $\varphi_{V_s}$  denotes the conjunction of the following axioms:

$$\bigwedge_{\epsilon \in Op} \left( \neg s_{\epsilon \preceq \epsilon} \rightarrow \bigwedge_{\substack{\epsilon' \in Op \\ \epsilon \neq \epsilon'}} \neg s_{\epsilon \preceq \epsilon'} \wedge \neg s_{\epsilon' \preceq \epsilon} \right) \quad (S1)$$

$$\bigwedge_{\epsilon \in Op} \bigwedge_{\substack{\epsilon' \in Op \\ \epsilon \neq \epsilon'}} \left( \neg s_{\epsilon \preceq \epsilon'} \vee \neg s_{\epsilon' \preceq \epsilon} \right) \quad (S2)$$

$$\bigwedge_{\epsilon \in Op} \bigwedge_{\substack{\epsilon' \in Op \\ \epsilon \neq \epsilon'}} \bigwedge_{\substack{\epsilon'' \in Op \\ \epsilon \neq \epsilon'' \\ \epsilon' \neq \epsilon''}} \left( s_{\epsilon \preceq \epsilon'} \wedge s_{\epsilon' \preceq \epsilon''} \rightarrow s_{\epsilon \preceq \epsilon''} \right) \quad (S3)$$



# $\exists\forall$ encoding of ES-INF planning problem $\langle \Sigma, Op, \alpha_G \rangle$

- $\varphi_{V_s}$  denotes the conjunction of the following axioms:

$$\bigwedge_{\epsilon \in Op} \bigwedge_{\substack{\epsilon' \in Op \\ \epsilon \neq \epsilon'}} \left( s_{\epsilon \preceq \epsilon} \wedge s_{\epsilon' \preceq \epsilon'} \rightarrow s_{\epsilon \preceq \epsilon'} \vee s_{\epsilon' \preceq \epsilon} \right) \quad (S4)$$

$$\bigwedge_{\epsilon \in Op} \bigwedge_{\epsilon' \in Op} \left( s_{\epsilon \preceq \epsilon'} \leftrightarrow \square_m s_{\epsilon \preceq \epsilon'} \right) \quad (S5)$$





## Dynamic extension (recall)

The following equivalences are valid in the class  $\mathbf{M}$ :

$$[+i\alpha]\alpha' \leftrightarrow \begin{cases} \top, & \text{if } \alpha' = \Delta_i\alpha, \\ \alpha', & \text{otherwise;} \end{cases}$$

$$[+i\alpha]\neg\varphi \leftrightarrow \neg[+i\alpha]\varphi;$$

$$[+i\alpha](\varphi_1 \wedge \varphi_2) \leftrightarrow [+i\alpha]\varphi_1 \wedge [+i\alpha]\varphi_2;$$

$$[+i\alpha](\varphi_1 \vee \varphi_2) \leftrightarrow [+i\alpha]\varphi_1 \vee [+i\alpha]\varphi_2;$$

$$[+i\alpha]\Box_m \alpha' \leftrightarrow \begin{cases} \Box_m(\alpha \rightarrow \alpha'), & \text{if } i = m, \\ \Box_m \alpha', & \text{otherwise;} \end{cases}$$

$$[+i\alpha]\Diamond_m \alpha' \leftrightarrow \begin{cases} \Diamond_m(\alpha \wedge \alpha'), & \text{if } i = m, \\ \Diamond_m \alpha', & \text{otherwise.} \end{cases}$$





# $\exists \forall$ encoding of ES-INF planning problem

- For  $\epsilon \in Op$ , we define the function  $\Pi_{\preceq \epsilon} : \mathcal{L}_{\text{Frag}} \longrightarrow \mathcal{L}_{\text{Frag}}$ .

$$\Pi_{\preceq \epsilon}(p) = p,$$

$$\Pi_{\preceq \epsilon}(\alpha) = \begin{cases} s_{\epsilon' \preceq \epsilon} \vee \Delta_m \alpha_{\epsilon'}, & \text{if } \exists \epsilon' \neq \epsilon : \alpha = \Delta_m \alpha_{\epsilon'}, \\ \alpha, & \text{otherwise;} \end{cases}$$

$$\Pi_{\preceq \epsilon}(\neg \varphi) = \neg \Pi_{\preceq \epsilon}(\varphi),$$

$$\Pi_{\preceq \epsilon}(\varphi_1 \wedge \varphi_2) = \Pi_{\preceq \epsilon}(\varphi_1) \wedge \Pi_{\preceq \epsilon}(\varphi_2),$$

$$\Pi_{\preceq \epsilon}(\varphi_1 \vee \varphi_2) = \Pi_{\preceq \epsilon}(\varphi_1) \vee \Pi_{\preceq \epsilon}(\varphi_2),$$

$$\Pi_{\preceq \epsilon}(\Box_m \alpha) = \Box_m \left( \left( \bigvee_{\substack{\epsilon' \in Op \\ \epsilon \neq \epsilon'}} s_{\epsilon' \preceq \epsilon} \wedge \neg \alpha_{\epsilon'} \right) \vee \alpha \right)$$

$$\Pi_{\preceq \epsilon}(\Diamond_m \alpha) = \Diamond_m \left( \bigwedge_{\substack{\epsilon' \in Op \\ \epsilon \neq \epsilon'}} (\neg s_{\epsilon' \preceq \epsilon} \vee \alpha_{\epsilon'}) \wedge \alpha \right)$$



## $\exists\forall$ encoding of ES-INF planning problem

Let  $F_{\langle \Sigma, Op, \alpha_G \rangle}$  be the reduction in  $\mathcal{L}_{Prop}$  of the following formula of  $\mathcal{L}_{Frag}$ :

$$\varphi_{V_s} \wedge \left( \left( \bigwedge_{\alpha \in \Sigma} \square_m \alpha \right) \rightarrow \bigwedge_{\epsilon \in Op} (s_{\epsilon \preceq \epsilon} \rightarrow \Pi_{\preceq \epsilon} (\mathcal{P}(\epsilon))) \wedge \varphi_P \right)$$

### Theorem

The ES-INF planning problem  $\langle \Sigma, Op, \alpha_G \rangle$  has a solution plan iff the following quantified boolean formula is true:

$$Q = \exists_{s_{\epsilon \preceq \epsilon'} \in V_s} \quad \forall_{p \in \text{Vars}(F_{\langle \Sigma, Op, \alpha_G \rangle}) \setminus V_s} \quad F_{\langle \Sigma, Op, \alpha_G \rangle}$$



# Part 2 - QBF encoding of cognitive planning and TouIST platform

TouIST Platform



# Motivation for a new tool

## Remark 1: existing languages are not intuitive

- ⊕ SAT solvers are very efficient
- ⊖ but with a “low level” input language

Formula	Solver input (DIMACS)
$a \rightarrow$	<code>p cnf 5 4</code>
$b \vee \neg(a \rightarrow c)$	<code>-1 5 3 0</code>
$\vee (c \rightarrow \neg a)$	<code>-1 -5 4 1 0</code>
	<code>-1 -5 4 -2 0</code>
	<code>-1 -3 -2 -1 0</code>





# Motivation for a new tool

## Remark 2: existing languages are not compact

- ⊕ SMT-LIB is much more expressive than DIMACS
- ⊖ but with a lack of compactity

Formula

$$\bigwedge_{i \in [1..1000]} x_i$$

SMT-LIB example

```
(assert
  (and x1
    (and x2
      (and x3
        (and x4
          (and x5
            (and x6
              ... x1000))))))))
```



# Motivation for a new tool

## Technical goal:

- ⊕ an intuitive language for modeling
- ⊕ a user-friendly graphical interface

## Academic goal:

- ⊕ teaching of logic
- ⊕ research (comparison of solvers, encodings)





# Principle of use for reasoning on a little example

## Example of the politician

If growth increases, then unemployment decreases ;

Now, growth decreases.

Then unemployment increases !

## Problem

Is the conclusion justified ?

$\{\text{growth} \rightarrow \neg\text{unemployment}, \neg\text{growth}\} \models \text{unemployment}$





# Principle of use for reasoning on a little example

$$\{\text{growth} \rightarrow \neg\text{unemployment}, \neg\text{growth}\} \models \text{unemployment}$$

The screenshot shows the Touist (english) v3.5.2 interface. On the left, a 'Connectors' panel lists logical operators:  $a \wedge b$ ,  $a \vee b$ ,  $\neg a$ ,  $a \oplus b$ ,  $a \Rightarrow b$ , and  $a \Leftrightarrow b$ . The main editor displays a list of four lines:

- 1 **growth**  $\Rightarrow$  **not** unemployment
- 2 **not** growth
- 3 **not** unemployment
- 4

The right pane shows the rendered formula in italics:

$$\text{growth} \Rightarrow \neg\text{unemployment}$$

$$\neg\text{growth}$$

$$\neg\text{unemployment}$$

At the bottom, there is a status bar showing '4:1' and a toolbar with icons for file operations, a 'Solve' button, and a dropdown menu currently set to 'SAT'.





# Principle of use for reasoning on a little example

Touist (english) v3.5.1

Results  Tr...  False  Others

Name ▲	Value
growth	False
unemployment	False

A model is found, then the reasoning **is not valid !**

$\{\text{growth} \rightarrow \neg\text{unemployment}, \neg\text{growth}\} \not\models \text{unemployment}$



## Implementation aspects



TouIST (Toulouse Integrated Satisfiability Tool)

Graphical interface for writing logical formulas  
in a **compact language** and using different **external solvers**.

- under MIT license, sources : [github.com/touist/touist](https://github.com/touist/touist)
- graphical interface (**Java**)
- translation to solver languages by a compiler (**OCaml**)
- compatible with **SAT, SMT and QBF**





# Implementation aspects

Compatible solvers :

Solver	Formula	ToulST input language
SAT	$a \wedge b \rightarrow c$	a and b => c
SMT (QF-LIA)	$(x + y - z \geq 4) \vee b$	(x + y - z >= 4) or a
SMT (QF-LRA)	$(x + y - z > 4.0) \rightarrow c$	(x + y - z > 4.0) => c
SMT (QF-IDL)	$(x - y > 2) \wedge c$	(x - y > 2) and c
SMT (QF-RDL)	$(x - y > 1.0) \wedge c$	(x - y > 1.0) and c
QBF	$\forall b. a \wedge b$	forall b: a and b

QF-LIA = Quantifier-Free Linear Integer Arithmetic

QF-LRA = Quantifier-Free Linear Rational Arithmetic

QF-IDL = Quantifier-Free Integer Difference Logic

QF-RDL = Quantifier-Free Rational Difference Logic



# Implementation aspects

ToulST input language :

## ■ variables and sets

```
$N = 5  
$Rows = [1..$N]  
$Countries = [France, Italy, TheNetherlands]
```

## ■ generalized connectors

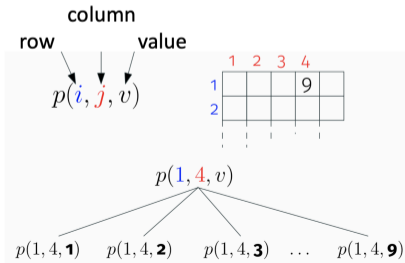
```
in(France) and in(Italy) and in(TheNetherlands)
```

```
bigand $c in $Countries:  
  in($c)  
end
```



# Solving Puzzles with TouIST (SAT)

	2			4	8	
1			8	2	3	7
			9	3	2	1
	4	8				1
			6	1	8	
7					9	3
	1	5	9	4		
4	2	3	6			9
3	7					5

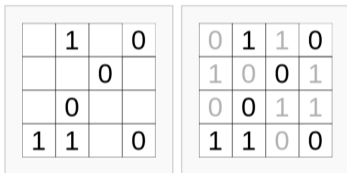


```

22 ;; at least one value per cell
23 bigand $i in [1..$N]:
24   bigand $j in [1..$N]:
25     bigor $k in [1..$N]:
26       p($i, $j, $k)
27     end
28   end
29 end
30
31 ;; at most one value per cell
32 bigand $i in [1..$N]:
33   bigand $j in [1..$N]:
34     bigand $k1 in [1..$N]:
35       bigand $k2 in [1..$N] when $k1 != $k2:
36         not p($i, $j, $k1) or not p($i, $j, $k2)
37       end
38     end
39   end
40 end
41
42 ;; at most once the same value per row
43 bigand $i in [1..$N]:
44   bigand $j1 in [1..$N]:
45     bigand $j2 in [1..$N] when $j1 != $j2:
46       bigand $k in [1..$N]:
47         not p($i, $j1, $k) or not p($i, $j2, $k)
48       end
49     end
50   end
51 end
52
53 ;; at most once the same value per column
54 bigand $i1 in [1..$N]:
55   bigand $i2 in [1..$N] when $i1 != $i2:
56     bigand $j in [1..$N]:
57       bigand $k in [1..$N]:
58         not p($i1, $j, $k) or not p($i2, $j, $k)
59       end
60     end
61   end
62 end
    
```



# Solving Puzzles with TouIST (SMT)



A compact SMT encoding  
with atoms of QF\_LIA  
(linear arithmetic on integers)

Touist (english) v3.5.2

```

26 ;; There is an equal number of 1s and 0s in each row
27 bigand $i in $N:
28   (x($i,1)+x($i,2)+x($i,3)+x($i,4)=2)
29   and (x(1,$i)+x(2,$i)+x(3,$i)+x(4,$i)=2)
30 end
31
32 ;; There is no more than two of either number adjacent
33 bigand $i,$j in [1..$NR-2],[1..$NC]:
34   x($i,$j)!=x($i+1,$j) or x($i+1,$j)!=x($i+2,$j)
35 end
36 bigand $i,$j in [1..$NR],[1..$NC-2]:
37   x($j,$i)!=x($j,$i+1) or x($j,$i+1)!=x($j,$i+2)
38 end
39
40 ;; There can be no identical rows or columns
41 bigand $i,$j in $N,$N when $i!=$j:
42   bigor $k in $N:
43     (x($i,$k)!=x($j,$k))
44   end
45   and
46   bigor $k in $N:
47     (x($k,$i)!=x($k,$j))
48   end
49 end
50
51

```

$$x_{i,j} \leftarrow [0]$$

$$\bigwedge (x_{i,j} = \text{value})$$

$$i,j,\text{value} \in [1..4],[1..4],x_{i,j}$$

$$\bigwedge ((x_{i,j} = 0) \vee (x_{i,j} = 1))$$

$$i,j \in \mathbb{N},\mathbb{N}$$

$$\bigwedge ((x_{i,1} + x_{i,2} + x_{i,3} + x_{i,4} = 2)$$

$$i \in \mathbb{N}$$

$$\bigwedge (x_{i,j} \neq x_{i+1,j} \vee x_{i,j} \neq x_{i+2,j})$$

$$i,j \in [1..NR-2],[1..NC]$$

$$\bigwedge (x_{j,i} \neq x_{j,i+1} \vee x_{j,i} \neq x_{j,i+2})$$

$$i,j \in [1..NR],[1..NC-2]$$

$$\bigwedge \left( \bigvee_{k \in \mathbb{N}} (x_{i,k} \neq x_{j,k}) \wedge \bigvee_{k \in \mathbb{N}, i \neq j} \right)$$

Solve QF\_LIA



# Solving an ES-INF planning problem with TouIST

- **SAT solver:** to verify the validity of a plan candidate

$$\left( \bigwedge_{\alpha \in \Sigma} \Box_m \alpha \right) \rightarrow \langle\langle \epsilon_1 \rangle\rangle \dots \langle\langle \epsilon_{k-1} \rangle\rangle \langle\langle \epsilon_k \rangle\rangle \Box_m \alpha_G$$

$$\mathcal{L}_{\text{Frag}} \xrightarrow{\text{nnf}} \mathcal{L}_{\text{Frag}}^{\text{NNF}} \xrightarrow{\text{tr}_1} \mathcal{L}_{\text{Mod}} \xrightarrow{\text{tr}_2} \mathcal{L}_{\text{Prop}}$$

- **QBF solver:** to extract a solution plan from  $\exists \forall$  encoding

$$Q = \exists_{s_{\epsilon \preceq \epsilon'} \in V_s} s_{\epsilon \preceq \epsilon'} \quad \forall_{p \in \text{Vars}(F_{\langle \Sigma, Op, \alpha_G \rangle}) \setminus V_s} p \quad F_{\langle \Sigma, Op, \alpha_G \rangle}$$



# Plan of the tutorial

- 1 Logical framework for cognitive planning (Emiliano)
- 2 QBF encoding of cognitive planning and ToulST platform (Frédéric)
- 3 **Implementation and applications (Jorge) NOW!**
  - Recommender system for sport activities
  - Conversational agent for motivational interviewing
- 4 Formal extension and implementation for collaborative gaming (Dominique & Jorge)

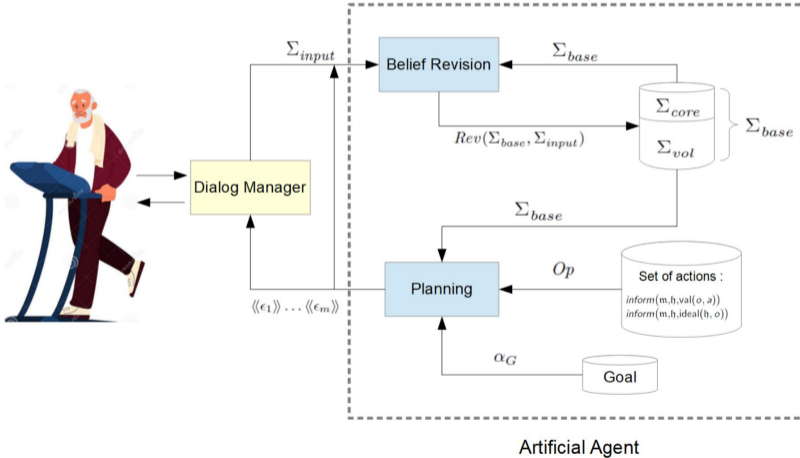




# Part 3 - Implementation and applications

Introduction

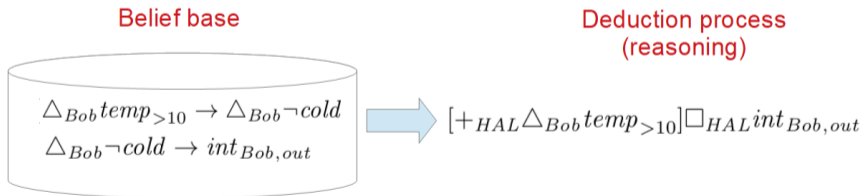
# IRIT System Architecture





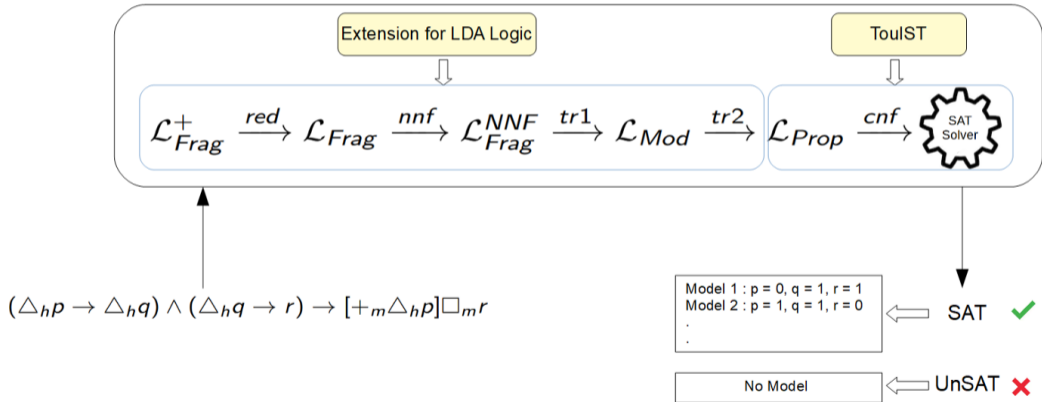
# Deduction process

We consider a conversational agent that has to interact with a human user in order to support her activity and to take care of her well-being. Specifically, HAL is an artificial companion which takes care of an elderly person called Bob and keeps him company. Bob has to do regular physical activity to be in good health.





# Checking satisfiability of formulas in $\mathcal{L}_{Frag}$





# Part 3 - Implementation and applications

Recommender system for sport activities



## Example 1: Artificial assistant

- Agent  $h$ : human user who has to choose a sport to practice
- Agent  $m$ : artificial assistant
- Agent  $m$ 's **goal**: agent  $h$  forms the intention to practice a sport
- **Solution plan**: sequence of speech acts by  $m$





## Example 1: Artificial assistant(cont.)

<i>Opt</i>	<i>Var</i>					
	<b>env</b>	<b>loc</b>	<b>soc</b>	<b>cost</b>	<b>dan</b>	<b>intens</b>
sw	<i>water</i>	<i>mixed</i>	<i>single</i>	<i>med</i>	<i>low</i>	<i>high</i>
ru	<i>land</i>	<i>outdoor</i>	<i>single</i>	<i>low</i>	<i>med</i>	<i>high</i>
hr	<i>land</i>	<i>outdoor</i>	<i>single</i>	<i>high</i>	<i>high</i>	<i>low</i>
te	<i>land</i>	<i>mixed</i>	<i>mixed</i>	<i>high</i>	<i>med</i>	<i>med</i>
so	<i>land</i>	<i>mixed</i>	<i>team</i>	<i>med</i>	<i>med</i>	<i>med</i>
yo	<i>land</i>	<i>mixed</i>	<i>single</i>	<i>med</i>	<i>low</i>	<i>low</i>
di	<i>water</i>	<i>mixed</i>	<i>single</i>	<i>high</i>	<i>high</i>	<i>low</i>
sq	<i>land</i>	<i>indoor</i>	<i>mixed</i>	<i>high</i>	<i>med</i>	<i>med</i>

**Table:** For every option  $o \in Opt$  and variable  $x \in Var$ , we denote by  $v_{o,x}$  the corresponding entry in the table.



# Example 1: Artificial assistant(cont.)

$$\alpha_1 \stackrel{\text{def}}{=} \bigwedge_{\substack{o \in \text{Opt} \\ x \in \text{Var} \\ v, v' \in \text{Val}_x: v \neq v'}} (\text{val}(o, x \mapsto v) \rightarrow \neg \text{val}(o, x \mapsto v')),$$

$$\alpha_2 \stackrel{\text{def}}{=} \bigwedge_{\substack{o \in \text{Opt} \\ x \in \text{Var} \\ v, v' \in \text{Val}_x: v \neq v'}} (\Delta_{\mathfrak{h}} \text{val}(o, x \mapsto v) \rightarrow \Delta_{\mathfrak{h}} \neg \text{val}(o, x \mapsto v')),$$

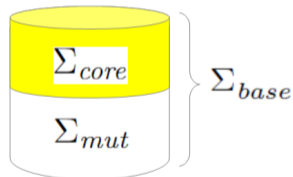
$$\alpha_3 \stackrel{\text{def}}{=} \bigwedge_{\Gamma, \Gamma' \in 2^{\text{Des}^*}: \Gamma \neq \Gamma'} (\text{des}(\mathfrak{h}, \Gamma) \rightarrow \neg \text{des}(\mathfrak{h}, \Gamma')),$$

$$\alpha_4 \stackrel{\text{def}}{=} \bigvee_{\Gamma \in 2^{\text{Des}^*}} \text{des}(\mathfrak{h}, \Gamma),$$

$$\alpha_5 \stackrel{\text{def}}{=} \bigwedge_{o \in \text{Opt}} (\text{ideal}(\mathfrak{h}, o) \leftrightarrow \bigvee_{\Gamma \in 2^{\text{Des}^*}} (\text{des}(\mathfrak{h}, \Gamma) \wedge \bigwedge_{\gamma \in \Gamma} f_{\text{comp}}(o, \gamma))),$$

$$\alpha_6 \stackrel{\text{def}}{=} \bigwedge_{o \in \text{Opt}} (\text{justif}(\mathfrak{h}, o) \leftrightarrow \bigvee_{\Gamma \in 2^{\text{Des}^*}} (\text{des}(\mathfrak{h}, \Gamma) \wedge \bigwedge_{\gamma \in \Gamma} f_{\text{comp}}^{\mathfrak{h}}(o, \gamma))).$$

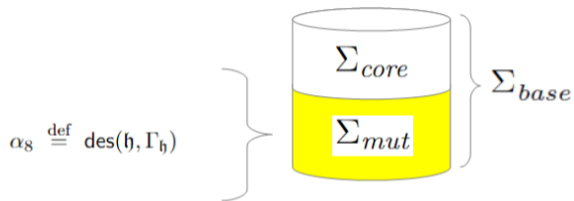
$$\alpha_7^{o,x} \stackrel{\text{def}}{=} \text{val}(o, x \mapsto v_{o,x}).$$







## Example 1: Artificial assistant(cont.)



$$\Gamma_h = \{\text{env} \mapsto \text{land}, \text{intens} \mapsto \text{med}, \sim \text{loc} \mapsto \text{indoor}, \\ [\text{cost} \mapsto \text{high}] \rightsquigarrow \text{soc} \mapsto \text{mixed}\}.$$

**Potential intention:**  $\text{potIntend}(h, o) \stackrel{\text{def}}{=}} \Delta_h \text{ideal}(h, o) \wedge \text{justif}(h, o)$





## Example 1: Artificial assistant(cont.)

### Operators:

$$Op = \{ \text{inform}(m, h, \text{val}(o, a)) : o \in Opt \text{ and } a \in Assign \} \cup \\ \{ \text{inform}(m, h, \text{ideal}(h, o)) : o \in Opt \}$$

### Executability preconditions:

$$\mathcal{P}(\text{inform}(m, h, \text{val}(o, a))) = \Box_m \left( \text{val}(o, a) \wedge \bigwedge_{v \in Val_{\mathbf{dan}}} (\text{val}(o, \mathbf{dan} \mapsto v) \rightarrow \Delta_h \text{val}(o, \mathbf{dan} \mapsto v)) \right)$$

if  $a \notin Assign_{\mathbf{dan}}$ ,

$$\mathcal{P}(\text{inform}(m, h, \text{val}(o, a))) = \Box_m \text{val}(o, a)$$

if  $a \in Assign_{\mathbf{dan}}$ ,

$$\mathcal{P}(\text{inform}(m, h, \text{ideal}(h, o))) = \Box_m (\text{ideal}(h, o) \wedge \text{justif}(h, o)).$$





## Example 1: Artificial assistant(cont.)

**Planning goal:**  $\alpha_G \stackrel{\text{def}}{=} \bigvee_{o \in Opt} \text{potIntend}(h, o)$

$\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4, \epsilon_5, \epsilon_6$  is a solution for the planning problem  $\langle \Sigma, Op, \alpha_G \rangle$  where:  
 $\Sigma = \{\alpha_1, \dots, \alpha_8\} \cup \{\alpha_7^{o,x} : o \in Opt \text{ and } x \in Var\}$  and

$\epsilon_1 \stackrel{\text{def}}{=} \text{inform}(m, h, \text{val}(te, \mathbf{dan} \mapsto med)),$

$\epsilon_2 \stackrel{\text{def}}{=} \text{inform}(m, h, \text{val}(te, \mathbf{env} \mapsto land)),$

$\epsilon_3 \stackrel{\text{def}}{=} \text{inform}(m, h, \text{val}(te, \mathbf{intens} \mapsto med)),$

$\epsilon_4 \stackrel{\text{def}}{=} \text{inform}(m, h, \text{val}(te, \mathbf{loc} \mapsto mixed)),$

$\epsilon_5 \stackrel{\text{def}}{=} \text{inform}(m, h, \text{val}(te, \mathbf{soc} \mapsto mixed)),$

$\epsilon_6 \stackrel{\text{def}}{=} \text{inform}(m, h, \text{ideal}(h, te)).$



A screenshot of a 'Chat Bot' window with a yellow border. The chat history shows the bot's introduction and questions about user preferences. A modal dialog titled 'Please, choose your preferences :' is open in the foreground. It has two columns: 'Variable' and 'Value'. Under 'Variable', there is a dropdown menu with 'intensity' selected. Under 'Value', there is a checkbox labeled 'Not' which is unchecked, and a vertical list of radio buttons with labels 'low', 'med', and 'high'. The 'med' option is currently selected. At the bottom of the dialog are 'Cancel' and 'Ok' buttons.

Chat Bot

Agent : Hi! How are you?, I am Botty your personal assistant  
Agent : I will help you to find the best sport based in your preferences  
Agent : Are you ready to start ?  
yes : Human

Agent : Great !, let's start by selecting your desires.  
Agent : Are you ready to choose your preferences ?  
yes : Human

Agent : You selected : env = land  
Agent : Do you want to continue adding more individual desires ?  
yes : Human

Please, choose your preferences :

Variable	Value
intensity	<input type="checkbox"/> Not
	<input checked="" type="radio"/> low
	<input type="radio"/> med
	<input type="radio"/> high

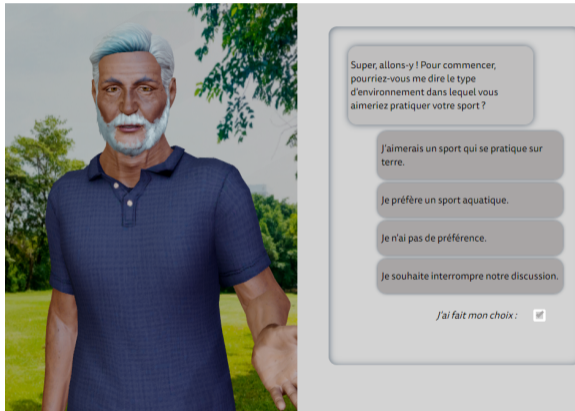
Cancel Ok

■ Source code:

<https://www.irit.fr/CoPains/software/>



# Web interface

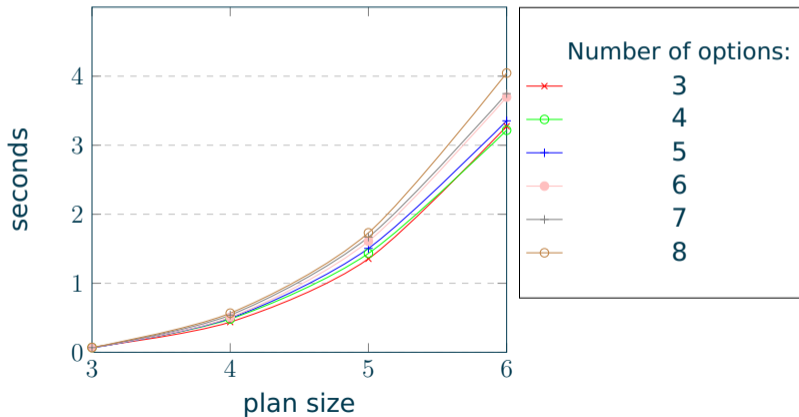


- Web avatar developed in collaboration with the company DAVI:

<https://cognitive-planning.schm.fr/>



Processing time used by the brute force approach:





## Experiments(cont.)

Processing time (in seconds) used by the brute force approach:

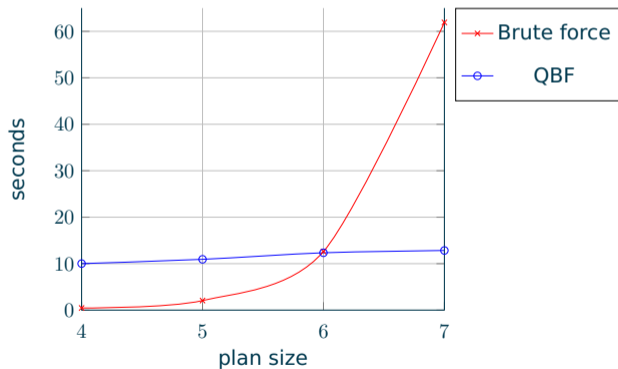
Plan size	Number of options (sports)					
	3	4	5	6	7	8
3	0.059	0.067	0.063	0.066	0.068	0.070
4	0.438	0.482	0.494	0.506	0.539	0.567
5	1.355	1.433	1.505	1.608	1.668	1.731
6	3.274	3.217	3.353	3.696	3.747	4.045





# Experiments(cont.)

Comparison in performance between the brute force and the QBF approaches:







# Experiments(cont.)

Comparison in performance between the brute force and the QBF approaches:

Size of the plan	Planning module processing time (seconds)	
	Brute force	QBF
4	0.464	10.015
5	2.059	10.936
6	12.610	12.343
7	61.920	12.837



## Conclusions

- Our implementation demonstrates that our NP-complete fragment and the cognitive planning problem formulated in this logic are suitable for real-world applications in the domain of HMI.
- We intend to include a setting parameter in the artificial agent in order to let the system select the most convenient approach (SAT or QBF) depending on the scenario.
- We plan to extend the implemented system by speech acts of type question to capture both sides of interaction, from agent  $m$  to agent  $h$  (handled by the actual implementation) and from agent  $h$  to agent  $m$ .



# Part 3 - Implementation and applications

Conversational agent for motivational interviewing



## Example 2: Virtual coaching

Agent  $m$  is a virtual coaching agent which has to motivate agent  $h$  to practice a physical activity.

⇒ Agent  $h$  has to become aware of the inconsistency between her current behavior and her desires

Desires : <i>DesAtm</i>	Conditions : <i>CondAtm</i>	Actions : <i>ActAtm</i>
<i>dr</i> : "h has dietary restrictions" <i>pw</i> : "h puts on weight" <i>lw</i> : "h loses weight" <i>at</i> : "h is attractive" <i>gh</i> : "h is in good health" <i>st</i> : "h is stressed"	<i>ow</i> : "h has an office work" <i>sl</i> : "h has a sedentary life style" <i>co</i> : "h is a commuter/time in traffic"	does( $h, ps^*$ )

\**ps* : "h practices sport"





## Example 2: Virtual coaching(cont.)

Agent  $m$ 's model of agent  $h$ 's cognitive state:

$$\alpha_1 \stackrel{\text{def}}{=} \bigwedge_{l \in Lit} \Delta_h \text{nec}(\{l\}, l),$$

$$\alpha_2 \stackrel{\text{def}}{=} \bigwedge_{l \in Lit} (\Delta_h \text{nec}(\emptyset, l) \leftrightarrow \Delta_h l),$$

$$\alpha_3 \stackrel{\text{def}}{=} \bigwedge_{l \in Lit, X, X' \in LitSet: X' \subseteq X} \left( \Delta_h \text{nec}(X, l) \rightarrow \left( \bigwedge_{l' \in X'} \Delta_h l' \rightarrow \Delta_h \text{nec}(X \setminus X', l) \right) \right),$$

$$\alpha_4 \stackrel{\text{def}}{=} \bigwedge_{l \in Lit, X, X' \in LitSet: X \subseteq X'} (\Delta_h \text{nec}(X, l) \rightarrow \Delta_h \text{nec}(X', l)),$$

$$\alpha_5 \stackrel{\text{def}}{=} \bigwedge_{l \in Lit} \left( (\text{des}(h, l) \leftrightarrow \Delta_h \text{des}(h, l)) \wedge (\neg \text{des}(h, l) \leftrightarrow \Delta_h \neg \text{des}(h, l)) \right),$$

$$\alpha_6 \stackrel{\text{def}}{=} \bigwedge_{a \in Act} \left( (\text{does}(h, a) \leftrightarrow \Delta_h \text{does}(h, a)) \wedge (\neg \text{does}(h, a) \leftrightarrow \Delta_h \neg \text{does}(h, a)) \right),$$



## Example 2: Virtual coaching(cont.)

$$\alpha_7 \stackrel{\text{def}}{=} \text{nec}(\{\neg dr, \neg pw, sl\}, \text{does}(h, ps)) \wedge \\ \text{nec}(\{at, \neg dr\}, \text{does}(h, ps)) \wedge \\ \text{nec}(\{sl, gh\}, \text{does}(h, ps)) \wedge \\ \text{nec}(\{gh\}, \neg st) \wedge \\ \text{nec}(\{co, ow\}, sl).$$

$$\alpha_G \stackrel{\text{def}}{=} \neg \text{does}(h, ps) \rightarrow \text{AwareIncon}(h, \neg \text{does}(h, ps)).$$

Plan computed by agent  $m$  :

Step	Informative planning problem	Interrogative planning problem
0		$?_{m,h} \text{does}(h, ps)$
1		$?_{m,h} \text{des}(h, gh), ?_{m,h} co, ?_{m,h} ow$
2	$!_{m,h}(\emptyset, \{co, ow\}, sl), !_{m,h}(\{gh\}, \{sl\}, \text{does}(h, ps))$	



## Example 2: Virtual coaching(cont.)

Speaker	Utterance	Speech act
m	Do you practice a sport regularly?	? <sub>m,h</sub> does(h,ps)
h	I don't	+ <sub>m</sub> ¬Δ <sub>h</sub> does(h,ps)
m	Do you wish to be in good health?	? <sub>m,h</sub> des(h,gh)
h	Yes	+ <sub>m</sub> Δ <sub>h</sub> des(h,gh)
m	Do you spend quite some time in the traffic everyday as a commuter?	? <sub>m,h</sub> co
h	Yes	+ <sub>m</sub> Δ <sub>h</sub> co
m	Do you have an office work?	? <sub>m,h</sub> ow
h	Yes	+ <sub>m</sub> Δ <sub>h</sub> ow
m	You spend quite some time in the traffic everyday as a commuter and you have an office work. Therefore, your life style is sedentary!	! <sub>m,h</sub> (∅, {co, ow}, sl)
m	Your life style is sedentary. Therefore, you will not satisfy your desire to be in good health unless you practice a sport regularly!	! <sub>m,h</sub> ({gh}, {sl}, does(h,ps))



# Plan of the tutorial

- 1 Logical framework for cognitive planning (Emiliano)
- 2 QBF encoding of cognitive planning and ToulST platform (Frédéric)
- 3 Implementation and applications (Jorge)
- 4 **Formal extension and implementation for collaborative gaming (Dominique & Jorge) NOW!**
  - Extended logical framework and encoding of the Yōkai game
  - General architecture and implementation





# Part 4 - Application to the Yōkai game

Introduction



# Why Yōkai game?

Because this game includes:

- reasoning about mental states (theory of mind),
- temporal reasoning,
- spatial reasoning.

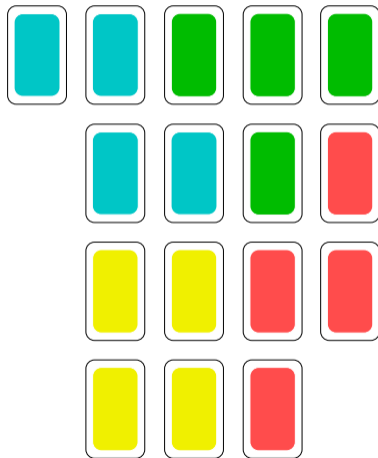


# Rules of Yōkai game [Griffon, 2019]

- 4 card colors and 4 cards of each color,
- 7 hints,
- initial state: shuffled and grouped cards face down,
- target end state (goal): group cards of the same color,
- 4 kinds of action:
  - to observe (the color of) a card,
  - to move a card (face down),
  - to active a hint,
  - to mark a card with a hint,
- constraint: communication outside of actions is not allowed.



## Example of a winning final configuration





## A round of play

At each round of play, each player execute 4 actions (no order constraint) :

- (to observe 1 card)  $\times 2$  ;
- to move 1 card (without creating a *separation*) ;
- to active 1 hint (x) or to mark 1 card with a hint about its color ( $\Rightarrow$  marked cards cannot be moved anymore).





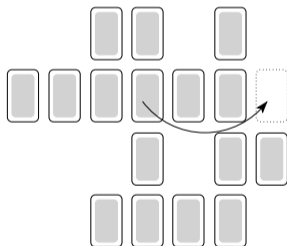
# What is a *separation* (between groups of cards)?

## Definition (separation)

There are at least two cards that cannot be linked by a “path”

## Definition (path)

A set of positions occupied by cards which are accessed only from their sides (NOT diagonally).

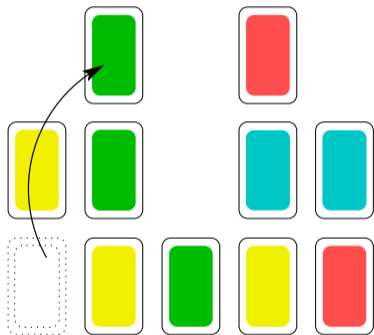




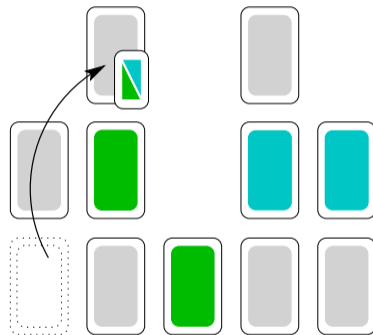
- an hint is a card containing 1 to 3 colors,
- “to mark a card” is to place a hint on this card,
- a card can only be marked with a hint including the color of the card,
- once marked, a card can no longer be observed, moved or marked.



# Example of game in progress ( $m$ plays against $h$ )



Cards of which  $m$  knows the color



Cards that  $m$  believes  $h$  knows the color of







## End of game

- 2 ways to complete the game:
  - either because a player declares the end of the game,
  - or because there is no more hint to activate or to mark a card.

### Reminder

- Each round, a hint must be activated or played (to mark a card),
- there are 7 hints,
- there are a maximum of 14 rounds of play (7 for each player).



# Part 4 - Application to the Yōkai game

Formal framework extension



## Temporal extension of explicit beliefs

We first define the language  $\mathcal{L}_0(Atm)$  by:

$$\alpha ::= \underbrace{p^t \mid \Delta_h^t \alpha \mid now^{\geq t}}_{\mathcal{L}_0^T(Atm)} \mid \neg \alpha \mid \alpha_1 \wedge \alpha_2 \mid \Delta_m \alpha$$

where :

- $p$  ranges over  $Atm$ ,
- $p^t$  is read “atomic proposition  $p$  is true at time  $t$ ”,
- $now^{\geq t}$  is read “the actual time of the game play is at least  $t$ ”.





## Implicit beliefs (no change)

Language  $\mathcal{L}(Atm)$  is defined by:

$$\varphi ::= \alpha \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \Box_m \alpha$$

Note: only formulas from  $\mathcal{L}_0$  can be in the scope of an  $\Box_m$  operator.





# Semantics (reminder & new things)

- State:  $S = (B, V)$  with
  - $B \subseteq \mathcal{L}_0$  is agent  $m$ 's belief base ,
  - $V \subseteq \mathcal{L}_0^T$  is the actual situation,
- for every  $t, t' \in \mathbb{N}$ :
  - $now^{\geq 0} \in V$ ,
  - if  $now^{\geq t} \in V$  and  $t' \leq t$  then  $now^{\geq t'} \in V$ ,
  - $now^{\geq t} \in V$  iff  $now^{\geq t} \in B$ .





## Semantics (new things)

Let  $S = (B, V) \in \mathcal{S}$ . Then, for every  $x \in \mathcal{L}_0^T(Atm)$ :

$$S \models x \iff x \in V,$$

$$S \models \neg\alpha \iff S \not\models \alpha,$$

$$S \models \alpha_1 \wedge \alpha_2 \iff S \models \alpha_1 \text{ and } S \models \alpha_2,$$

$$S \models \Delta_m \alpha \iff \alpha \in B.$$





# Dynamic extension

- Language  $\mathcal{L}^+(Atm)$  is defined as follows:

$$\varphi ::= \alpha \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \Box_m \alpha \mid [+^t_m \alpha]\varphi,$$

- $[+^t_m \alpha]\varphi$  is read “ $\varphi$  holds after agent  $m$  has privately learned that  $\alpha$  and that the current time is at least  $t$ ”.
- $(S, Cxt) \models [+^t_m \alpha]\varphi \iff (S^{+^t_m \alpha}, Cxt) \models \varphi$  with

$$S^{+^t_m \alpha} = (B^{+^t_m \alpha}, V^{+^t_m \alpha}),$$

$$V^{+^t_m \alpha} = V \cup \{now^{\geq t'} : t' \leq t\},$$

$$B^{+^t_m \alpha} = B \cup \{\alpha\} \cup \{now^{\geq t'} : t' \leq t\}.$$



# Theorems

- Checking satisfiability of  $\mathcal{L}(Atm)$  formulas in the class  $\mathbf{M}$  is an NP-complete problem.
- Checking satisfiability of  $\mathcal{L}^+(Atm)$  formulas in the class  $\mathbf{M}$  is an NP-complete problem.
- Let  $\varphi \in \mathcal{L}^+(Atm)$  and let  $\Sigma \subset \mathcal{L}_0(Atm)$  be finite. Then
 
$$\Sigma \models_{\mathbf{M}} \varphi \text{ if and only if } \models_{\mathbf{M}} \bigwedge_{\alpha \in \Sigma} \Box_m \alpha \rightarrow \varphi.$$

(Logical consequence problem with a finite set of premises can be reduced to the satisfiability problem.)







## Action selection (rappel)

- **Belief expansion events:**  $Act_m = \{+^t_m \alpha : \alpha \in \mathcal{L}_0, t \in \mathbb{N}\}$
- **Executability preconditions:**  $\mathcal{P}(\epsilon)$  for  $\epsilon \in Act_m$
- **Successful occurrence of an event:**  $\langle\langle +^t_m \alpha \rangle\rangle \varphi \stackrel{\text{def}}{=} \mathcal{P}(+^t_m \alpha) \wedge [+^t_m \alpha] \varphi$
- **Action selection problem (ASP):**  $\langle \Sigma, Op, \alpha_G \rangle$  where:
  - $\Sigma \subset \mathcal{L}_0(Atm)$  (m's belief base),
  - $Op \subset Act_m$  (m's action repertoire),
  - $\alpha_G \in \mathcal{L}_0(Atm)$  (m's goal).
- **Solution to ASP:**  $\epsilon \in Op$  such that  $\Sigma \models_M \langle\langle \epsilon \rangle\rangle \Box_m \alpha_G$ .





## Update, then revision

- Belief update :  $m$ 's explicit beliefs evolve from a time  $t$  to  $t + 1$ .
- Belief revision :  $m$  learns a new fact and adds it to its belief base.





## Beliefs update

- $Upd(\Sigma_c, \Sigma_h, \Sigma_f) = (\Sigma'_c, \Sigma'_h, \Sigma'_f)$  if and only if:
  - $\Sigma'_c = \Sigma_c$ ,
  - $\Sigma'_h = incrt(\Sigma_h)$ ,
  - $\Sigma'_f = incrt(\Sigma_f) \cup \{now^{\geq 0}\}$ .
  
- where for each finite  $X \subseteq \mathcal{L}_0$ ,  $incrt(X) = \{incrt(\alpha) : \alpha \in X\}$  such that:
  - $incrt(p^t) = p^{t+1}$
  - $incrt(\Delta_h^t \alpha) = \Delta_h^{t+1} incrt(\alpha)$
  - $incrt(\Delta_m \alpha) = \Delta_m incrt(\alpha)$
  - $incrt(now^{\geq t}) = now^{\geq t+1}$
  - $incrt(\neg \alpha) = \neg incrt(\alpha)$
  - $incrt(\alpha_1 \wedge \alpha_2) = incrt(\alpha_1) \wedge incrt(\alpha_2)$ .



## Belief revision (reminder)

- $Rev(\Sigma_c, \Sigma_h, \Sigma_f, \Sigma_{input}) =$ 
  - $(\Sigma_c, \Sigma_h, \Sigma_f)$  if  $\Sigma_c \cup \Sigma_{input}$  is not propositionally consistent ;
  - $(\Sigma'_c, \Sigma'_h, \Sigma'_f)$  otherwise, with  $\Sigma'_c = \Sigma_c$ ,  $\Sigma'_f = \bigcap_{X \in MCS(\Sigma_c, \Sigma_f, \Sigma_{input})} X$ ,  
 $\Sigma'_h = \bigcap_{X \in MCS(\Sigma_c \cup \Sigma'_f, \Sigma_h, \emptyset)} X$ .
  
- So:
  - $\Sigma_c$  is not modified,
  - $\Sigma_{input}$  is added to  $\Sigma_f$  (iff  $\Sigma_{input}$  and  $\Sigma_c$  are consistent),
  - belief revision satisfies minimal change for factual information.





# Part 4 - Application to the Yōkai game

Logical modeling of Yōkai



Let be the following sets:

$$TIME = \{0, 1, \dots, end\} \text{ where } end = 56$$

$$TIME^* = TIME \setminus \{0\}$$

$$GRID = \{1, \dots, 32\} \times \{1, \dots, 32\},$$

$$IPOS = \{(l, c) \in GRID : l, c \in \{15, \dots, 18\}\},$$

$$COLORS = \{r, g, b, y\},$$

$$HINTS = 2^{COLORS} \setminus \{\{\}, \{r, g, b, y\}\},$$

$$CARDS = \{1, \dots, 16\}$$

$$CARDS^n = \{X \in 2^{CARDS} : |X| = n\} \text{ with } n \in \mathbb{N}$$





# Atomic propositions

- $col_{x,c}^t$ : card  $x$  is of color  $c$  at time  $t$ ,
- $pos_{x,p}^t$ : card  $x$  is at position  $p$  at time  $t$ ,
- $active_h^t$ : hint  $h$  is enabled at time  $t$ ,
- $mark_{x,h}^t$ : card  $x$  is marked with hint  $h$  at time  $t$ ,
- $legMov_{x,p}^t$ : to move card  $x$  to position  $p$  at time  $t$  is legal.





# Initial factual belief base

$$\Sigma_f^W = \{now^{\geq 0}\} \cup \bigcup_{x \in CARDS} \{pos_{x,\sigma(x)}^0\} \cup \bigcup_{\substack{x \in CARDS \\ c \in COLORS}} \{\neg \Delta_m col_{x,c}^0\} \cup$$

$$\bigcup_{h \in HINTS} \{\neg active_h^0\} \cup \bigcup_{\substack{x \in CARDS \\ h \in HINTS}} \{\neg mark_{x,h}^0\}$$

$$\Sigma_f^h = \bigcup_{\alpha \in \Sigma_f^W} \{\Delta_h^0 \alpha\} \cup \bigcup_{\substack{x \in CARDS \\ c \in COLORS}} \{\neg \Delta_h^0 col_{x,c}^0\}$$

$$\Sigma_f = \Sigma_f^W \cup \Sigma_f^h$$







# Initial hypothetical belief base

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

## Example

1)  $\Sigma_h \supseteq \{ \text{mark}_{1,\{g,r\}}^0 \rightarrow \neg \text{col}_{1,b}^0, \text{mark}_{1,\{g,r\}}^0 \rightarrow \neg \text{col}_{1,y}^0 \}$ .

2)  $\text{mark}_{1,\{g,r\}}^0 \in \Sigma_f$ .

So, we can deduce that  $\Box_m ((\text{col}_{1,g}^0 \vee \text{col}_{1,r}^0) \wedge \neg \text{col}_{1,b}^0 \wedge \neg \text{col}_{1,y}^0)$  holds.



# Core belief base (integrity constraints)

## Rules of the game:

$$\Sigma_c^{ic} = \bigcup_{t \in TIME} \left\{ \bigwedge_{x \in CARDS} \bigvee_{p \in GRID} pos_{x,p}^t, \right. \quad (ICP1)$$

$$\bigwedge_{\substack{x \in CARDS \\ p, p' \in GRID: p \neq p'}} \neg(pos_{x,p}^t \wedge pos_{x,p'}^t), \quad (ICP2)$$

$$\bigwedge_{\substack{p \in GRID \\ x, x' \in CARDS: x \neq x'}} \neg(pos_{x,p}^t \wedge pos_{x',p}^t), \quad (ICP3)$$

$$\dots \left. \right\}$$



## Core belief base (frame axioms)

- Description of what does not change:

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

$$\text{colFA}_X^t \stackrel{\text{def}}{=} \bigwedge_{\substack{x \in \text{CARDS} \setminus X \\ c \in \text{COLORS}}} \left( (\text{col}_{x,c}^t \leftrightarrow \text{col}_{x,c}^{t-1}) \wedge (\neg \Delta_m \text{col}_{x,c}^t \leftrightarrow \neg \Delta_m \text{col}_{x,c}^{t-1}) \right)$$

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



# Core belief base (successor state axioms)

## ■ Computation of a future state:

$$\Sigma_c^{ssa} = \bigcup_{t \in \text{TIME}^*} \left\{ \bigwedge_{\substack{x \in \text{CARDS} \\ p \in \text{GRID}}} (pos_{x,p}^t \wedge \neg pos_{x,p}^{t-1} \rightarrow posFA_{\{x\}}^t \wedge colFA_{\emptyset}^t \wedge hintFA_{\emptyset}^t), \right. \quad (\text{SSA4})$$

$$\bigwedge_{\substack{x \in \text{CARDS} \\ c \in \text{COLORS}}} (col_{x,c}^t \wedge \neg \Delta_m col_{x,c}^{t-1} \rightarrow posFA_{\emptyset}^t \wedge colFA_{\{x\}}^t \wedge hintFA_{\emptyset}^t), \quad (\text{SSA5})$$

$$\bigwedge_{h \in \text{HINTS}} (active_h^t \wedge \neg active_h^{t-1} \rightarrow posFA_{\emptyset}^t \wedge colFA_{\emptyset}^t \wedge hintFA_{\{h\}}^t), \quad (\text{SSA6})$$

$$\left. \bigwedge_{\substack{x \in \text{CARDS} \\ h \in \text{HINTS}}} (mark_{x,h}^t \wedge \neg mark_{x,h}^{t-1} \rightarrow posFA_{\emptyset}^t \wedge colFA_{\emptyset}^t \wedge hintFA_{\{h\}}^t) \right\} \quad (\text{SSA7})$$

 **Core belief base**

## ■ Finally:

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



# Action repertoire

- $ACT \subseteq EVT$  of agent  $m$  is defined as follows:

$$+_m^{col_{x,c}^t} \stackrel{\text{def}}{=} +_m^t \left( col_{x,c}^t \wedge \Delta_m col_{x,c}^t \wedge \left( \Delta_h^t \bigvee_{c' \in COLORS} \Delta_m col_{x,c'}^t \right) \wedge \Delta_h^t now^{\geq t} \right)$$

$$+_m^{pos_{x,p}^t} \stackrel{\text{def}}{=} +_m^t \left( pos_{x,p}^t \wedge \Delta_m pos_{x,p}^t \wedge \Delta_h^t pos_{x,p}^t \wedge \Delta_h^t now^{\geq t} \right)$$

$$+_m^{actHint_h^t} \stackrel{\text{def}}{=} +_m^t \left( active_h^t \wedge \Delta_m active_h^t \wedge \Delta_h^t active_h^t \wedge \Delta_h^t now^{\geq t} \right)$$

$$+_m^{markHint_{x,h}^t} \stackrel{\text{def}}{=} +_m^t \left( mark_{x,h}^t \wedge \Delta_m mark_{x,h}^t \wedge \Delta_h^t mark_{x,h}^t \wedge \Delta_h^t now^{\geq t} \right)$$



# Actions preconditions

$$\mathcal{P}(+_m^{\text{col}}{}_{x,c}^{t+1}) \stackrel{\text{def}}{=} \text{now}^=t \wedge \square_m \bigwedge_{h \in \text{HINTS}} \neg \text{mark}_{x,h}^t$$

$$\mathcal{P}(+_m^{\text{pos}}{}_{x,p}^{t+1}) \stackrel{\text{def}}{=} \text{now}^=t \wedge \square_m \bigwedge_{h \in \text{HINTS}} \neg \text{mark}_{x,h}^t \wedge \square_m \text{legMov}_{x,p}^t$$

$$\mathcal{P}(+_m^{\text{actHint}}{}_h^{t+1}) \stackrel{\text{def}}{=} \text{now}^=t$$

$$\mathcal{P}(+_m^{\text{markHint}}{}_{x,h}^{t+1}) \stackrel{\text{def}}{=} \text{now}^=t \wedge \square_m \text{active}_h^t \wedge \bigvee_{c \in h} \square_m \text{col}_{x,c}^t \wedge \square_m \bigwedge_{\substack{h' \in \text{HINTS} \\ h' \neq h}} \neg \text{mark}_{x,h'}^t$$





## Goals modeling (observe a card around $X$ )

- goal = motivational attitude for action,
- $X \subseteq CARDS : 1 \leq |X| \leq 3$ , a set of cards of the same color,

$$\alpha_{obsArroundCards(X)}^t \stackrel{\text{def}}{=} \bigvee_{\substack{x' \in CARDS \\ x' \notin X}} \left( \left( \bigwedge_{c \in COLORS} \neg \Delta_m col_{x',c}^t \right) \wedge \right. \\ \left. nbgCards_{X \cup \{x'\}}^t \wedge \left( \bigvee_{c \in COLORS} \Delta_m col_{x',c}^{t+1} \right) \right)$$

$$\mathcal{P}_g(\alpha_{obsArroundCards(X)}^t) \stackrel{\text{def}}{=} nbgCards_X^t$$







# Goals modeling (observe a card whose color is unknown)

Let  $x \in CARDS$ :

$$\alpha_{obsUnknownColorCard}^t(x) \stackrel{\text{def}}{=} \bigvee_{c \in COLORS} \Delta_m col_{x,c}^{t+1}$$

$$\mathcal{P}_g(\alpha_{obsUnknownColorCard}^t(x)) \stackrel{\text{def}}{=} \bigwedge_{c \in COLORS} \neg \Delta_m col_{x,c}^t$$





# Goals modeling (observe a card randomly)

- Default goal for moving ( $x \in CARDS$ ):

$$\alpha_{obsRandomly(x)}^t \stackrel{\text{def}}{=} \bigvee_{c \in COLORS} \Delta_m col_{x,c}^{t+1}$$

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





## How to manage the different goal?

- With meta algorithms generating an ordered list of goals
- $\text{getSubsetsGtLeq}(X, \tau_m, \tau_M) = \{X' \in 2^X : \tau_m < |X'| \leq \tau_M\}$

```

1: function tryToObserve( $S, \alpha_g^t$ )
2:    $success \leftarrow \text{false}$ 
3:   while  $S \neq \emptyset$  && ! $success$  do
4:      $x \leftarrow \text{pop}(S)$ 
5:     if  $\mathcal{P}_g(\alpha_{g(x)}^t)$  then
6:        $success \leftarrow \text{checkPlan}(\alpha_{g(x)}^t)$ 
7:     end if
8:   end while
9:   return  $success$ 

```

```

1: ▷ Parameters computation for  $\alpha_{obsArroundCards(X)}^t$ 
2:  $S \leftarrow \{\}$ 
3: for all  $c \in COLORS$  do
4:    $X_c \leftarrow \{x \in CARDS : \Delta_m col_{x,c}^t \in \Sigma_f\}$ 
5:    $S \leftarrow S \cup \text{getSubsetsGtLeq}(X_c, 0, |X_c|)$ 
6: end for
7:  $S' \leftarrow \text{sortByDecreasingSize}(S)$ 
8:   ▷  $\forall X_i, X_{i+1} \in S', |X_i| \geq |X_{i+1}|$ 

```



## How to manage the different goal?

- 1: **if** !tryToObserve( $S'$ ,  $\alpha_{obsArroundCards}^t$ ) **then**
- 2:      $X \leftarrow \{x \in CARDS : \exists c \in COLORS, \Delta_m col_{x,c}^t \in \Sigma_f\}$
- 3:      $X' \leftarrow CARDS \setminus X$                    ▷ The set of cards whose color is unknown
- 4:     **if** !tryToObserve( $X'$ ,  $\alpha_{obsUnknownColorCard}^t$ ) **then**
- 5:         tryToObserve( $CARDS$ ,  $\alpha_{obsRandomly}^t$ )
- 6:     **end if**
- 7: **end if**



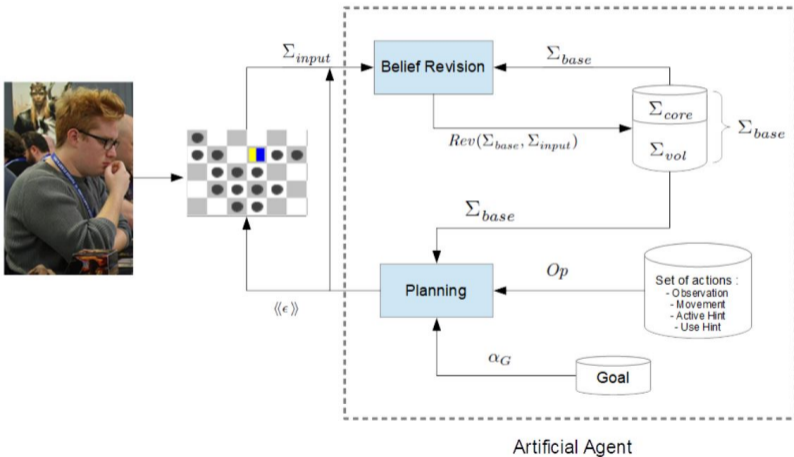


# Part 4 - Application to the Yōkai game

Implementation

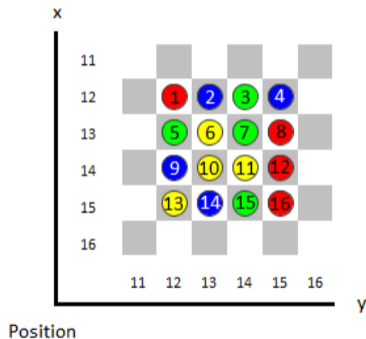


# System Architecture





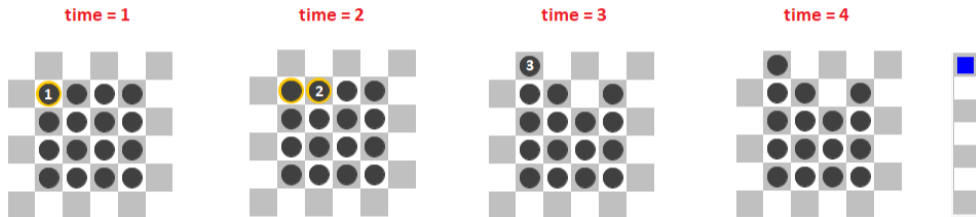
# Cards distribution



- Source code:  
<https://github.com/iritlab/yokai>



# Agent $m$ 's first round



- 1:  $col\_1\_R\_t4$  and  $tm\_col\_1\_R\_t4$  and ( $th\_tm\_col\_1\_G\_t4$  or  $th\_tm\_col\_1\_Y\_t4$  or  $th\_tm\_col\_1\_B\_t4$  or  $th\_tm\_col\_1\_R\_t4$ )
- 2:  $col\_2\_B\_t4$  and  $tm\_col\_2\_B\_t4$  and ( $th\_tm\_col\_2\_G\_t4$  or  $th\_tm\_col\_2\_Y\_t4$  or  $th\_tm\_col\_2\_B\_t4$  or  $th\_tm\_col\_2\_R\_t4$ )
- 3:  $pos\_3\_p\_11\_12\_t4$  and  $tm\_pos\_3\_p\_11\_12\_t4$  and  $th\_pos\_3\_p\_11\_12\_t4$
- 4:  $act\_1\_t4$  and  $tm\_act\_1\_t4$  and  $th\_act\_1\_t4$

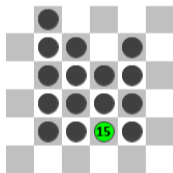
$\Sigma_m$  at time point 4



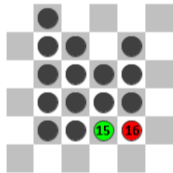


# Agent $h$ 's first round

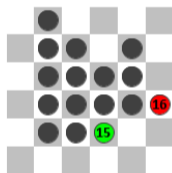
time = 5



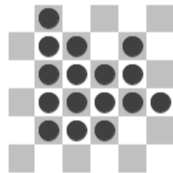
time = 6



time = 7



time = 8



5: (th\_col\_15\_R\_t8 or th\_col\_15\_G\_t8 or th\_col\_15\_Y\_t8 or th\_col\_15\_B\_t8)

6: (th\_col\_16\_R\_t8 or th\_col\_16\_G\_t8 or th\_col\_16\_Y\_t8 or th\_col\_16\_B\_t8)

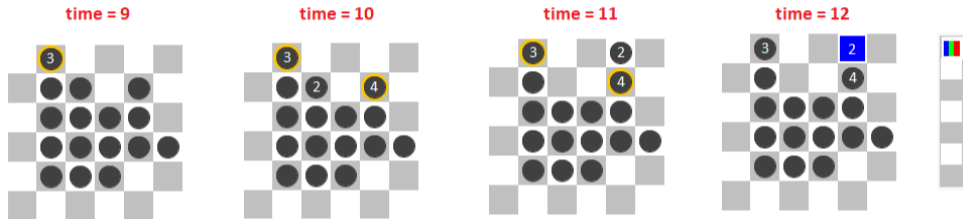
7: pos\_16\_p\_14\_16\_t8 and tm\_pos\_16\_p\_14\_16\_t8 and th\_pos\_16\_p\_14\_16\_t8

8: act\_6\_t8 and tm\_act\_6\_t8 and th\_act\_6\_t8

$\Sigma_m$  at time point 8



# Grouping cards



2: `col_2_B_t10` and `tm_col_2_B_t10` and (`th_tm_col_2_G_t10` or `th_tm_col_2_Y_t10` or `th_tm_col_2_B_t10` or `th_tm_col_2_R_t10`)

9: `col_3_G_t10` and `tm_col_3_G_t10` and (`th_tm_col_3_G_t10` or `th_tm_col_3_Y_t10` or `th_tm_col_3_B_t10` or `th_tm_col_3_R_t10`)

10: `col_4_B_t10` and `tm_col_4_B_t10` and (`th_tm_col_4_G_t10` or `th_tm_col_4_Y_t10` or `th_tm_col_4_B_t10` or `th_tm_col_4_R_t10`)



# Action selection

```
(not ([m](
not tm_col_5_G_t10 and
not tm_col_6_Y_t10 and
.
col_1_R_t10 and tm_col_1_R_t10 and (th_tm_col_1_G_t10 or th_tm_col_1_Y_t10 or th_tm_col_1_B_t10 or th_tm_col_1_R_t10) and
col_2_B_t10 and tm_col_2_B_t10 and (th_tm_col_2_G_t10 or th_tm_col_2_Y_t10 or th_tm_col_2_B_t10 or th_tm_col_2_R_t10) and
act_1_t10 and tm_act_1_t10 and th_act_1_t10 and
.
;;SSA11
((pos_2_p_11_15_t11 and not pos_2_p_11_15_t10) =>
((pos_1_p_12_12_t11 <=> pos_1_p_12_12_t10) and ((pos_3_p_11_12_t11 <=> pos_3_p_11_12_t10) and
((pos_4_p_12_15_t11 <=> pos_4_p_12_15_t10) and ((pos_5_p_13_12_t11 <=> pos_5_p_13_12_t10) and
((pos_6_p_13_13_t11 <=> pos_6_p_13_13_t10))))))
.
=> ([m] now_t10 and [m] (not mark_2_1_t10 and not mark_2_2_t10 and
not mark_2_3_t10 and not mark_2_4_t10 and not mark_2_5_t10 and
not mark_2_6_t10 and not mark_2_7_t10) and
plus(pos_2_p_11_15_t11 and tm_pos_2_p_11_15_t11 and th_pos_2_p_11_15_t11,
[m]((pos_2_p_11_15_t11 and pos_4_p_12_15_t11) or (pos_2_p_12_14_t11 and pos_4_p_12_15_t11) or
(pos_2_p_12_16_t11 and pos_4_p_12_15_t11) or (pos_4_p_11_13_t11 and pos_2_p_12_13_t11) or
(pos_4_p_12_14_t11 and pos_2_p_12_13_t11))))))
```

Planning formula for grouping blue cards 2 and 4



# Belief revision

## Integrity constraint ICP2

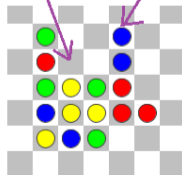
```

.
.
(not (pos_2_p_12_13_t11 and pos_2_p_12_11_t11) and
(not (pos_2_p_12_13_t11 and pos_2_p_10_12_t11) and
(not (pos_2_p_12_13_t11 and pos_2_p_11_11_t11) and
(not (pos_2_p_12_13_t11 and pos_2_p_11_13_t11) and
.
.
(not (pos_2_p_12_13_t11 and pos_2_p_11_15_t11) and
(not (pos_2_p_12_13_t11 and pos_2_p_12_14_t11) and
(not (pos_2_p_12_13_t11 and pos_2_p_12_16_t11) and
(not (pos_2_p_12_13_t11 and pos_2_p_13_11_t11) and

```

Initial position

Final position

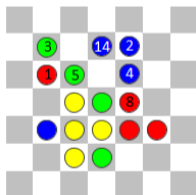




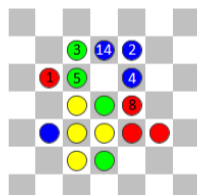
# Hierarchy of goals

*Sets of cards known by the machine :*

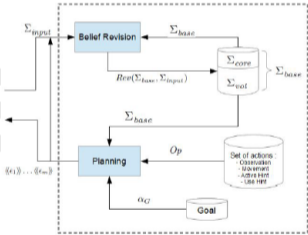
1	B	2	4	14
2	R	1	8	
3	G	3	5	
4	B	4	14	
5	B	2	4	
6	B	2	14	
7	Y	6		



*Human round at time t*

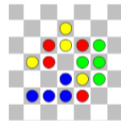


*Machine round at time t+3*



Artificial Agent

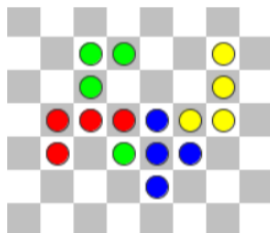
h - m



h - h



# Experiments



Case 01 - Failed game



Case 02 - Failed game

**Figure: Two different scenarios when players lost the game**





## Experiments(cont.)

Experiment set	Game	h-m		h-h	
		# 4-y-groups	Score	# 4-y-groups	Score
1	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	

**Table: Total # 4-y-groups in experiment set 1**





## Experiments(cont.)

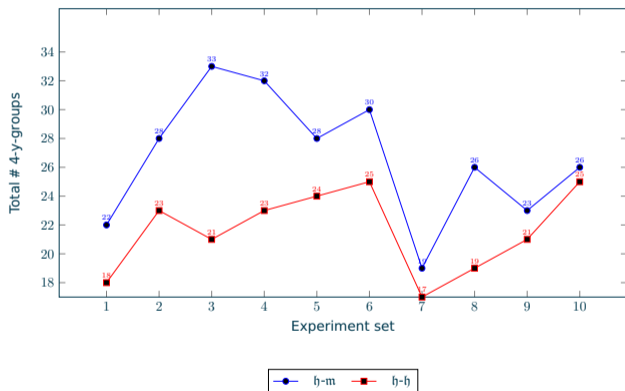
Config. type	Experiment set									
	1	2	3	4	5	6	7	8	9	10
h-m	22	28	33	32	28	30	19	26	23	26
h-h	18	23	21	23	24	25	17	19	21	25

**Table: Total # 4-y-groups by experiment set**





# Experiments(cont.)



**Figure: Total # 4-y-groups by experiment set**



Griffon, J. (2019).

Règles du jeu de Yōkai.

[https://www.play-in.com/pdf/rules\\_games/yokai\\_regles\\_fr.pdf](https://www.play-in.com/pdf/rules_games/yokai_regles_fr.pdf).