

An Implemented System for Cognitive Planning

Jorge Fernandez¹^a, Dominique Longin²^b, Emiliano Lorini²^c and Frédéric Maris¹^d

¹*IRIT, Toulouse University, France*

²*IRIT, CNRS, Toulouse University, France*

Keywords: Knowledge Representation, Epistemic Logic, Satisfiability, Epistemic Planning, Cognitive Planning, Persuasion.

Abstract: We present a system that implements a framework for cognitive planning. The system allows us to represent and reason about the beliefs, desires and intentions of other agents using an NP-fragment of a multiagent epistemic logic. The system has three components: the belief revision, the planning and the translator modules. They work in an integrated way to firstly capture new information about the world, secondly to plan a sequence of speech acts aimed at achieving a persuasive goal and, finally, to verify satisfiability of the formulas generated at each step of the process. We illustrate how our system can be used to implement a persuasive artificial agent interacting with a human user.

1 INTRODUCTION

Automated planning is at the center of AI research with a variety of applications ranging from control traffic and robotics to logistics and services. Epistemic planning extends automated planning incorporating notions of knowledge and beliefs (Bolander and Andersen, 2011; Löwe et al., 2011; Kominis and Geffner, 2015; Muise et al., 2015; Cooper et al., 2021). Cognitive planning is a generalization of epistemic planning, where the goal to be achieved is not only a belief state but a cognitive state of a target including not only beliefs but also intentions. Moreover, we are particularly interested in persuasive goals of the planning agent, aimed at influencing another agent's beliefs and intentions.


The increasing number of applications in social robotics, social networks, virtual assistants together with sentiment analysis techniques allow us to collect data related to humans' beliefs and intentions. In (Akimoto, 2019) a framework for modeling mental attitudes of an agent, based on her narratives, is proposed. In addition, cognitive models can be used to predict agents' decision-making by taking psychological factors like motivation and emotions into account (Prezenski et al., 2017). Nonetheless, few approaches


exist which leverage this information about humans' cognitive states for changing their attitudes and behaviors through persuasion.


Our work aims to fill this gap by introducing a system¹ based on a simple framework detailed in (Fernandez et al., 2021) in which we can represent an agent's cognitive state in a compact way, reason about it and planning a sequence of speech acts aimed at changing it. Our approach is based on an epistemic logic introduced in (Lorini, 2018; Lorini, 2020), which allows us to represent an agent's explicit beliefs, as the information in the agent's belief base, and the agent's implicit beliefs, as the information which is deducible from the agent's belief base. Given that the satisfiability problem for the full logic is PSPACE-hard, we focus on an NP-fragment that makes the logic suitable for implementing real-world applications.


The core components of the system are the belief revision, the planning and the translator modules. The formulas representing the rules and constraints for a specific problem domain are loaded into the system. We encode these rules using the NP-fragment presented in (Fernandez et al., 2021). The system takes this information as the initial state and some actions — which are of type speech act — to build a plan that leads to the goal. An important feature is that actions have preconditions that impose constraints on their execution order. We illustrate the implementa-

¹<https://github.com/CognitivePlanning/sw>

^a <https://orcid.org/0000-0001-9328-1670>

^b <https://orcid.org/0000-0002-3138-2262>

^c <https://orcid.org/0000-0002-7014-6756>

^d <https://orcid.org/0000-0002-1084-1669>

tion of our system in a human-machine interaction (HMI) scenario in which an artificial agent has to persuade a human agent to practice a sport based on her preferences.

2 A LANGUAGE FOR EXPLICIT AND IMPLICIT BELIEF

This section describes the basics of the Logic of Doxastic Attitudes (LDA) introduced in (Lorini, 2018; Lorini, 2020). It is a multiagent epistemic logic which supports reasoning about explicit and implicit beliefs. Assume a countably infinite set of atomic propositions Atm and a finite set of agents $Agt = \{1, \dots, n\}$. We define the language in two steps.

We first define the language $\mathcal{L}_0(Atm, Agt)$ by the following grammar in Backus-Naur Form (BNF):

$$\alpha ::= p \mid \neg\alpha \mid \alpha_1 \wedge \alpha_2 \mid \alpha_1 \vee \alpha_2 \mid \Delta_i\alpha,$$

where p ranges over Atm and i ranges over Agt . $\mathcal{L}_0(Atm, Agt)$ is the language for representing agents' explicit beliefs. The formula $\Delta_i\alpha$ is read “ i explicitly believes that α ”. The language $\mathcal{L}(Atm, Agt)$ extends the language $\mathcal{L}_0(Atm, Agt)$ by modal operators of implicit belief and is defined by the following grammar:

$$\varphi ::= \alpha \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \Box_i\varphi \mid \Diamond_i\varphi,$$

where α ranges over $\mathcal{L}_0(Atm, Agt)$ and i ranges over Agt . For notational convenience we write \mathcal{L}_0 instead of $\mathcal{L}_0(Atm, Agt)$ and \mathcal{L} instead of $\mathcal{L}(Atm, Agt)$, when the context is unambiguous. The formula $\Box_i\varphi$ is read “ i implicitly believes that φ ” and $\Diamond_i\varphi$ is read “ φ is compatible (or consistent) with i 's explicit beliefs”. The other Boolean constructions \top , \perp , \rightarrow and \leftrightarrow are defined in the standard way.

The language is interpreted with respect to a formal semantics using belief bases whose details are given in (Lorini, 2018; Lorini, 2020). Checking satisfiability of \mathcal{L} formulas relative to this semantics is a PSPACE-hard problem. For that reason, in (Fernandez et al., 2021), we looked for an interesting NP-fragment of \mathcal{L} that we called \mathcal{L}_{Frag}^+ :

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

where α ranges over \mathcal{L}_0 and m is a special agent in Agt called the ‘machine’. In \mathcal{L}_{Frag}^+ , all agents have explicit beliefs but only agent m has implicit beliefs. Therefore, formulas including nesting implicit belief operators are not allowed (e.g., $\Box_m\neg\Box_m p$ is not a well-formed formula). Moreover the latter are restricted to \mathcal{L}_0 formulas of type α .

Agent m is the artificial planning agent. In order to represent agents' belief dynamics, language \mathcal{L}_{Frag}^+

$$\mathcal{L}_{Frag}^+ \xrightarrow{red} \mathcal{L}_{Frag} \xrightarrow{mf} \mathcal{L}_{Frag}^{NNF} \xrightarrow{tr_1} \mathcal{L}_{Mod} \xrightarrow{tr_2} \mathcal{L}_{Prop}$$

Figure 1: Summary of reduction process.

is extended by belief expansion operators. Such an extension will allow us to represent the actions of the planning agent in the cognitive planning problem. Specifically, we introduce the following language \mathcal{L}_{Frag}^+ :

$$\varphi ::= \alpha \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \Box_m\alpha \mid \Diamond_m\alpha \mid [+_i\alpha]\varphi,$$

where α ranges over \mathcal{L}_0 and i ranges over Agt . The formula $[+_i\alpha]\varphi$ is read “ φ holds after agent i has privately expanded her belief base with α ”. Event of type $+_i\alpha$ are generically called informative actions.

3 COGNITIVE PLANNING

The planning problem in the context of the logic \mathcal{L}_{Frag}^+ is to find a sequence of informative actions for agent m of type $+_m\alpha$ which guarantees that agent m will knowingly achieve its goal. Let $Act_m = \{+_m\alpha : \alpha \in \mathcal{L}_0\}$ be agent m 's set of informative actions and let the elements of Act_m be noted $\varepsilon, \varepsilon', \dots$. Agent m 's informative actions have executability preconditions that are specified by the following function: $\mathcal{P} : Act_m \rightarrow \mathcal{L}_{Frag}$. So, we can define the following operator of successful occurrence of an informative action:

$$\langle\langle\varepsilon\rangle\rangle\varphi \stackrel{\text{def}}{=} \mathcal{P}(\varepsilon) \wedge [\varepsilon]\varphi$$

with $\varepsilon \in Act_m$. The formula $\langle\langle\varepsilon\rangle\rangle\varphi$ has to be read “agent m 's informative action ε can take place and φ holds after its occurrence”.

Informative actions of type ‘speech act’ are of interest here. In particular, we consider speech acts of type ‘to inform’, where m is assumed to be the speaker and $j \in Agt$ such that $j \neq m$ is assumed to be the hearer. We define the speech act “agent m informs agent j that α ” as follows:

$$inform(m, j, \alpha) \stackrel{\text{def}}{=} +_m \Delta_j \alpha.$$

In (Fernandez et al., 2021) the planning problem is defined as a tuple $\langle \Sigma, Op, \alpha_G \rangle$ where:

- $\Sigma \subset \mathcal{L}_0$ is a finite set of agent m 's available information,
- $Op \subset Act_m$ is a finite set of agent m 's operators,
- $\alpha_G \in \mathcal{L}_0$ is agent m 's goal.

The planning problem has a solution if the formula $\neg((\bigwedge_{\alpha \in \Sigma} \Box_m \alpha) \rightarrow \langle\langle\varepsilon_1\rangle\rangle \dots \langle\langle\varepsilon_k\rangle\rangle \Box_m \alpha_G)$ is unsatisfiable. Checking plan existence for a \mathcal{L}_{Frag}^+ -planning problem is in $NP^{NP} = \Sigma_2^P$.

In Algorithm 1, $plan_{\mathcal{L}_{Frag}^+}[k, i]$ (line 8) is the i candidate plan of size k , generated from the following elements: the belief base, the i subset in the set of combinations C of size k from Op and the goal.

Algorithm 1: Cognitive planning.

Data: Σ_{base} , Op , α_G
Result: Plan

```

1 Begin
2   Function combinations ( $Op, k$ )
3      $C =$  All subsets of size  $k$  from  $Op$ ;
4     return  $C$ 
5   Function generatePlans ( $k$ )
6      $C \leftarrow$  combinations( $Op, k$ );
7     for  $i \leftarrow 1$  to  $|C|$  do
8        $plan_{\mathcal{L}_{Frag}^+}[k, i] =$ 
9          $\neg(\Box_m \Sigma_{base} \rightarrow [+_m C[i]] \Box_m \alpha_G)$ 
10       $plan_{\mathcal{L}_{Prop}}[k, i] =$ 
11        Translator( $plan_{\mathcal{L}_{Frag}^+}[k, i]$ )
12      if TouIST( $plan_{\mathcal{L}_{Prop}}[k, i]$ ) =
13        UnSAT then
14          Print "Plan  $i$  of size  $k$  is
15            valid" ;
16          success  $\leftarrow$  true;
17          return
18        end if
19      end for
20      return
21 Main
22  $k = 1$  ;
23 success  $\leftarrow$  false;
24 while ( $k \leq |Op|$  || not(success)) do
25   generatePlans( $k$ );
26    $k++$ ;
27 end while
28 if not(success) then
29   Print "Plan not found" ;
30 end if
31 exit

```

4 IMPLEMENTATION

The functionality of our integrated system for cognitive planning is defined by the use case presented in Figure 2.

Its two core modules are belief revision and cognitive planning which work in an integrated way. Firstly, the belief revision module reads the input, coming from the human through the graphical user interface (GUI) for dialog, and verifies that this input does not contradict the core beliefs stored in the belief

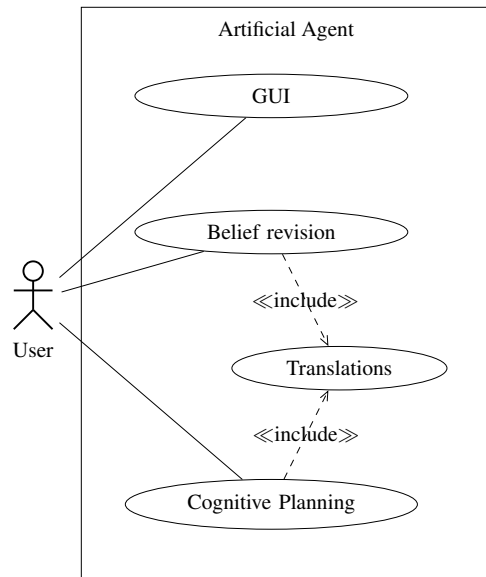


Figure 2: Use Case Artificial Agent.

base. Core beliefs are fundamental beliefs that never change and are distinguished from volatile beliefs that can change due to revision. If the input is in contradiction with the core beliefs, then the input is rejected and the belief base is not updated. On the contrary, if the input is not in contradiction with the core beliefs, then the belief base is revised using a maximal consistent subset (MCS) approach whereby the input has priority over the old volatile beliefs.

Secondly, the planning module reads the initial state, the set of actions, and the goal and starts to generate candidate plans of different size, starting with size equal to one. During this phase, the planning module calls the translator module which converts the \mathcal{L}_{Frag}^+ planning formula into its equivalent in propositional logic, following the sequence of reductions detailed in Figure 1. After the reduction process performed by the translator, the planning module executes the SAT encoding tool TouIST (Fernandez et al., 2020) to verify the validity of the propositional formula. TouIST will encode the formula in CNF format and send it to MiniSAT (this solver is set by default in the application) for checking satisfiability. TouIST can work with external solvers that accept standardized DIMACS as input language. Figure 3 shows the proposed system architecture.

In order to probe the potential of our implemented system for cognitive planning we applied it to a HMI scenario detailed in (Fernandez et al., 2021). In this scenario agent m is the artificial assistant of the human agent h . Agent h has to choose a sport to practice since her doctor recommended her to do a regular physical activity to be in good health. Agent m 's aim

Table 1: Variable assignments. For every option $o \in Opt$ and variable $x \in Var$, we denote by $v_{o,x}$ the corresponding entry in the table. For instance, we have $v_{sw,env} = water$.

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

is to help agent h to make the right choice, given her actual beliefs and desires.

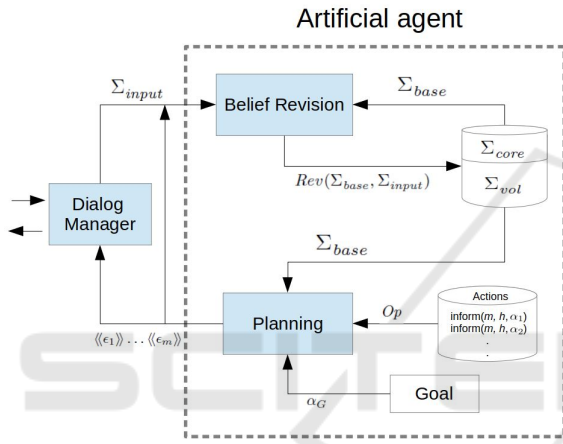


Figure 3: System architecture.

In order to set the initial belief base, agent m has to be provided with information about the possible options that the user can choose (Opt) and their properties (Var). For each pair (Opt, Var) we have a valuation Val . In this example, we suppose that Opt includes the following eight elements: swimming (sw), running (ru), horse riding (hr), tennis (te), soccer (so), yoga (yo), diving (di) and squash (sq). Moreover, there are exactly six variables in Var which are used to classify the available options: environment (**env**), location (**loc**), sociality (**soc**), cost (**cost**), dangerousness (**dan**) and intensity (**intens**). The variable assignments are shown in Table 1.

Formulas representing the rules and constraints are loaded as part of agent m 's belief base. For example, the implementation of the formula representing the fact that agent h explicitly believes that a sport cannot have two different values for a given property is formalized as follows:

$$\bigwedge_{\substack{o \in Opt \\ x \in Var \\ v_1, v_2 \in Val_x: v_1 \neq v_2}} \left(\Delta_h \text{val}(o, x \mapsto v_1) \rightarrow \Delta_h \neg \text{val}(o, x \mapsto v_2) \right)$$

The syntax for writing the formulas is based on the TouIST language, with the extension of the modal operators for explicit and implicit belief. For example, we use $\{h\}$ for representing Δ_h . Similarly we use $[m]$ for \Box_m .

```
bigand
  $o, $x, $v1, $v2
  in $Opt, $Var, $Val($x), $Val($x)
  when $v1 != $v2:
  {h}val($o, ass($x, $v1)) =>
    {h}not val($o, ass($x, $v2))
end
```

Thus, this syntax allows us to represent functions like the one included in the next formula, which states that an option o is ideal for agent h if and only if the option satisfies all agent h 's desires:

$$\bigwedge_{o \in Opt} \left(\text{ideal}(h, o) \leftrightarrow \bigvee_{\Gamma \in 2^{Des}} \left(\text{des}(h, \Gamma) \wedge \bigwedge_{\gamma \in \Gamma} f_{comp}(o, \gamma) \right) \right)$$

The function f_{comp} specifies, for every option $o \in Opt$ and possible desire $\gamma \in Des$, the condition guaranteeing that o satisfies (or, complies with) γ :

$$\begin{aligned} f_{comp}(o, a) &= \text{val}(o, a), \\ f_{comp}(o, \sim a) &= \neg \text{val}(o, a), \\ f_{comp}(o, [d_1, \dots, d_k] \rightsquigarrow d) &= \neg f_{comp}(o, d_1) \vee \dots \vee \neg f_{comp}(o, d_k) \vee f_{comp}(o, d). \end{aligned}$$

The full implementation of the formula with the function f_{comp} included, requires the capture of the human's desires which together with the set of rules and constraints are used to generate the machine's belief base.

```
$n1 = 2
$n2 = 1
$n3 = 1

$Delta0 = [
  "ass(env, land)",
  "ass(intens, med)",
  "not ass(loc, indoor)",
  "ass(cost, high) => ass(soc, mixed)"
]

$Delta0_1(1) = ["ass(env, land)"]
$Delta0_1(2) = ["ass(intens, med)"]
$Delta0_2(1) = ["not ass(loc, indoor)"]
$Delta0_2(1,1) = ["ass(loc, indoor)"]
$Delta0_3(1) = ["ass(cost, high) =>
  ass(soc, mixed)"]
$Delta0_3(1,1) = ["ass(cost, high)"]
$Delta0_3(1,2) = ["ass(soc, mixed)"]

$Opt = [sw, ru, te, hr, so, yo, di, sq]
$Var = [env, loc, soc, cost, danger, intens]
```

```

$Val(env)      = [land,water]
$Val(loc)      = [indoor, outdoor, mixed]
$Val(soc)      = [single, team, mixed]
$Val(cost)     = [low, med, high]
$Val(danger)  = [low, med, high]
$Val(intens)   = [low, med, high]
.
.
.
bigand
  $o in $Opt :
  ideal(h,$o) <=>
    ((bigand
      $d0,$i,$e
      in $Delta0, [1..$n1],
      $Delta0_1($i)
      when $d0 in $Delta0_1($i):
      val($o,$e)
    end) and
    (bigand
      $d0,$i,$e
      in $Delta0, [1..$n2],
      $Delta0_2($i,1)
      when $d0 in $Delta0_2($i):
      not val($o,$e)
    end) and
    (bigand
      $d0,$i,$p,$c
      in $Delta0, [1..$n3],
      $Delta0_3($i,1),
      $Delta0_3($i,2)
      when $d0 in $Delta0_3($i):
      not val($o,$p) or val($o,$c)
    end))
end

```

The set of human desires is represented by $\$Delta0$ in the previous syntax. The counters $\$n1, \$n2, \$n3$ specify the number of positive desires, negative desires and conditional desires respectively. We conceive a positive desire as the human expressing a valuation for a variable assignment from Table 1 (e.g., environment is land). A negative desire is a negative valuation for a variable assignment. Finally, a conditional desire is a conditional valuation between variable assignments (e.g., if the cost is high then sociality level should be mixed).

Similarly, the goal to be achieved by the planning agent is captured by the following formula:

$$\alpha_G \stackrel{\text{def}}{=} \bigvee_{o \in Opt} \text{potIntend}(h, o).$$

Moreover, we suppose that, for agent h to have a potential intention to choose option o , denoted by $\text{potIntend}(h, o)$, she must have a justified belief that o is an ideal option for her:

$$\text{potIntend}(h, o) \stackrel{\text{def}}{=} \Delta_h \text{ideal}(h, o) \wedge \text{justif}(h, o).$$

The latter is defined using the same syntax and in our case is expressed by the next formula:

```

bigor
  $o in $Opt :
  {h}ideal(h,$o) and justif(h,$o)
end

```

The set of actions are generated from Table 1. For instance, $\text{inform}(m, h, \text{val_so_ass_env_land})$ is an informative action. It is interpreted as the speech act used by agent m to inform agent h that the valuation of the property:environment for the option:soccer is land. In order to help agent h to select an activity, agent m also needs information about h 's desires. This information is gathered by agent m during its interaction with agent h . The interaction interface between h and m is shown in Figure 4. The belief revision module is called after each agent h 's feedback and it restores consistency of the agent m 's belief base, in case the incoming information is inconsistent with agent m 's pre-existent beliefs. In the ex-

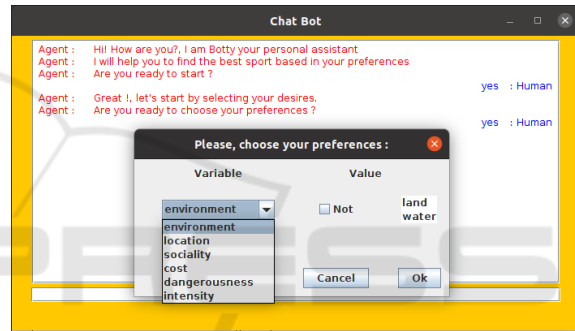


Figure 4: Collecting agent h 's preferences.

ample, agent h would like to practice a land activity, with medium intensity, which is not exclusively indoor, and which can be practiced both in single and team mode, if its cost is high. The next rule for precondition states that agent h must be informed by agent m about the dangerousness level of a sport, before presenting other properties for an option. For $a \notin \text{Assign}_{\text{dan}}$:

$$\mathcal{P}(\text{inform}(m, h, \text{val}(o, a))) = \square_m (\text{val}(o, a) \wedge \bigwedge_{v \in \text{Val}_{\text{dan}}} (\text{val}(o, \text{dan} \mapsto v) \rightarrow \Delta_h \text{val}(o, \text{dan} \mapsto v)))$$

In the next lines we illustrate how the precondition is assigned by the planning module together with its $+\text{m}\alpha$ operator in order to specify the successful occurrence of an informative action:

```

[m] ((val_te_ass_intens_med) and
      (val_te_ass_danger_med =>
        {h}val_te_ass_danger_med)) and
      plus({h}val_te_ass_intens_med...

```

The planning module generates plans with the elements contained in the action file. It starts with plans

of length 1, and enters in a loop. At each interaction the planning module asks the SAT solver to verify whether the plan allows to achieve the goal. If no plan of length k is found, the program will increase the counter in one and look for a plan of length $k + 1$. An example of an abstract plan generated by the planning module is:

```
plus({h})(val_te_ass_danger_med)
plus({h})(val_te_ass_intens_med)
plus({h})(val_te_ass_soc_mixed)
plus({h})(val_te_ass_loc_mixed)
plus({h})(val_te_ass_env_land)
plus({h})(ideal_h_te)
```

The order of speech acts is determined by the pre-conditions. Specifically, the planning module informs firstly about the dangerousness level of the sport. Secondly, it provides explanation of why the user’s desires are satisfied. Finally, it indicates the ideal sport for the user, in this case tennis.

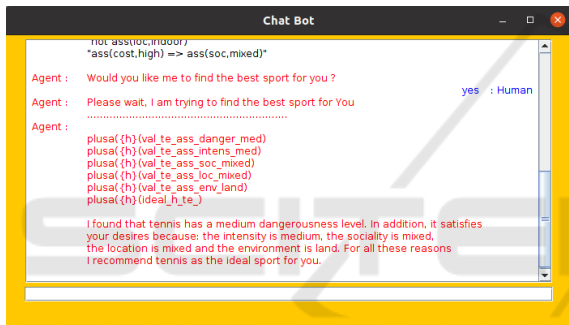


Figure 5: Plan shown by the chatbot to the human.

The chatbot writes both the sequence of speech acts and its translation into natural language expressions. We decided to display the abstract plan in the GUI, as shown in Figure 5, for illustrative purposes oriented to demonstrate how the GUI transforms it into natural language using a simple function. The abstract plan will not be displayed by the GUI in the end-user version of the system.

5 EXPERIMENTS

In this section, we present the experiments conducted in order to test the cognitive planning system in the scenario described in the previous section. The experiment was devoted to evaluate the performance of the planning module in integration with the belief revision module. The GUI was not used during the test, therefore the procedure was carried out on command line mode.

In order to perform the test we generate firstly a set of desires of the human in different input files. These

input files are processed by the belief revision module sequentially in order to generate the volatile side of the belief base. Secondly, the translator module is called to generate the initial state and the goal. Finally, the initial state, the set of actions (repertoire of speech acts) and the goal are used to call the planning module.

The set of options and variables described in Table 1 were used to test the performance of the system, expanding the table in the number of sports available. Similarly, we vary the number of the human’s desires. Figure 6 shows the results of the computation. The data plotted in the previous graph are shown in table 2.

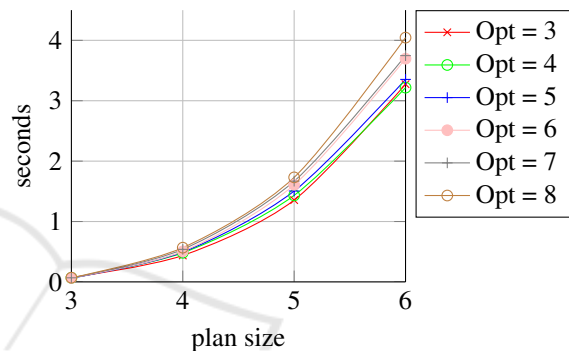


Figure 6: Processing time (in seconds) based on the number of options.

Table 2: Processing time (in seconds) to achieve a plan based of the number of Options.

Plan size	Number of Options (Opt)					
	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

The experiments were conducted using an Ubuntu 64 bits linux virtual machine running on a core i7 processor with 8 gigabytes RAM. The belief revision and cognitive planning module were implemented in Ocaml version 4.10.0 and the chatbot interface was programmed in Java openjdk version 1.8.0 with swing components. The data files containing the belief base, actions, goal and plan were stored as plain text files.

6 DISCUSSION

The architecture presented in Figure 3 works as an integrated system. All the processes, interfaces and

exchange of data between the modules are working according to the definition of the use cases displayed in Figure 2.

The system dialogue capability is limited for the moment. The human agent communicates her desires to the machine, and the latter computes the most suitable plan. There is no feedback from the human after the sequence of speech acts performed by the machine.

The effectiveness of the computation is polynomial with respect to the set of actions. Although the algorithm for choosing the correct plan uses a brute force technique, the experiments demonstrate that in order to verify the validity of a single candidate plan, the planning module takes around 66 ms on average. The reason for choosing a brute force approach was to allow the algorithm to be the most general as possible. However, it would be possible to include a heuristic to improve the performance of the general process. For example, the planning module could consider the size of the input (based on the human's set of desires) as the initial size of the plan. Thus, the planner will generate plans of that size at least. This prevents the planner from spending time to generate candidate plans of smaller size than the number of human's desires. In addition, an optimization can be included in the algorithm if we add a mechanism for giving priorities to certain types of actions. For example, the actions which are stated as preconditions should be prioritized to be included between the first sets of combinations to be tested by the planning module.

Despite the fact that the SAT encoding is efficient for solving the planning problem, we still need to generate one formula per candidate plan, which is time-consuming, especially for the translation process and the interface with the external tool TouIST. We want to explore the possibility of using a QBF (quantified boolean formulas) encoding of the planning problem which will allow us to generate one single formula for evaluating all possible candidate plans. In this case, the preconditions are assigned if and only if there exists a plan that satisfies the goal. This alternative approach will allow us compare the efficiency of QBF solvers against the SAT-based method in solving our planning problem.

7 CONCLUSION

Our implementation demonstrates that the NP-complete epistemic logic presented in (Fernandez et al., 2021) and the cognitive planning problem formulated in this logic are suitable for real-world applications in the domain of human-machine interaction.

In future work, 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 . We expect to apply the same framework to a joint activity scenario of type cooperative boardgame (Bard et al., 2019; Longin et al., 2020) involving the human and the machine in which they have to exchange information and collaborate in order to achieve a common goal.

We also plan to combine our implementation of cognitive planning with machine learning and data mining techniques, as presented in (Krzywicki et al., 2016), in order to extract information about the human user from real data. In addition, 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 think that the SAT approach could be better when the set of actions is not so big, while the QBF approach will turn out to be well-suited for handling a large repertoire of speech acts.

Last but not least, we intend to compare our framework for cognitive planning with approaches to epistemic planning closely related to ours (Muise et al., 2015; Kominis and Geffner, 2015; Le et al., 2018). None of these approaches presents a modular architecture or an integration of planning and belief revision. On the contrary, we have built our system in a modular way by designing the different components, including planning and belief revision, with the interfaces that are necessary for the system working in an integrated way. This feature allows the scalability of the system. In fact, we are expecting to replace the GUI with a more advanced web interface and add a security module for granting access to the system in a multi-user environment without altering the rest of the modules.

ACKNOWLEDGEMENTS

This work is supported by the ANR project CoPains ("Cognitive Planning in Persuasive Multimodal Communication"). Support from the ANR-3IA Artificial and Natural Intelligence Toulouse Institute is also acknowledged.

REFERENCES

- Akimoto, T. (2019). Narrative structure in the mind: Translating genette's narrative discourse theory into a cog-

- nitive system. *Cognitive Systems Research*, 58:342–350.
- Bard, N., Foerster, J. N., Chandar, S., Burch, N., Lanctot, M., Song, H. F., Parisotto, E., Dumoulin, V., Moitra, S., Hughes, E., Dunning, I., Mourad, S., Larochelle, H., Bellemare, M. G., and Bowling, M. (2019). The hanabi challenge: A new frontier for AI research. *CoRR*, abs/1902.00506.
- Bolander, T. and Andersen, M. B. (2011). Epistemic planning for single- and multi-agent systems. *Journal of Applied Non-Classical Logics*, 21(1):9–34.
- Cooper, M. C., Herzig, A., Maffre, F., Maris, F., Perrotin, E., and Régnier, P. (2021). A lightweight epistemic logic and its application to planning. *Artificial Intelligence*, 298:103437.
- Fernandez, J., Gasquet, O., Herzig, A., Longin, D., Lorini, E., Maris, F., and Régnier, P. (2020). TouIST: a friendly language for propositional logic and more. In Bessiere, C., editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 5240–5242. International Joint Conferences on Artificial Intelligence Organization. Demos.
- Fernandez, J., Longin, D., Lorini, E., and Maris, F. (2021). A simple framework for cognitive planning. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21)*. AAAI Press.
- Kominis, F. and Geffner, H. (2015). Beliefs in multiagent planning: from one agent to many. In Brafman, R. I., Domshlak, C., Haslum, P., and Zilberstein, S., editors, *Proceedings of the 25th International Conference on Automated Planning and Scheduling (ICAPS 2015)*, pages 147–155. AAAI Press.
- Krzywicki, A., Wobcke, W., Bain, M., Calvo Martinez, J., and Compton, P. (2016). Data mining for building knowledge bases: Techniques, architectures and applications. *The Knowledge Engineering Review*, -1:1–27.
- Le, T., Fabiano, F., Son, T. C., and Pontelli, E. (2018). EFP and PG-EFP: epistemic forward search planners in multi-agent domains. In *Proceedings of the Twenty-Eighth International Conference on Automated Planning and Scheduling, ICAPS 2018*. AAAI Press.
- Longin, D., Lorini, E., and Maris, F. (2020). Beliefs, time and space: A language for the yōkai board game. In Uchiya, T., Bai, Q., and Marsá-Maestre, I., editors, *PRIMA 2020: Principles and Practice of Multi-Agent Systems - 23rd International Conference, Nagoya, Japan, November 18-20, 2020, Proceedings*, volume 12568 of *Lecture Notes in Computer Science*, pages 386–393. Springer.
- Lorini, E. (2018). In praise of belief bases: Doing epistemic logic without possible worlds. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, pages 1915–1922. AAAI Press.
- Lorini, E. (2020). Rethinking epistemic logic with belief bases. *Artificial Intelligence*, 282.
- Löwe, B., Pacuit, E., and Witzel, A. (2011). DEL planning and some tractable cases. In *Proceedings of the 3rd International International Workshop on Logic, Rationality and Interaction (LORI 2011)*, pages 179–192. Springer Berlin Heidelberg.
- Muise, C., Belle, V., Felli, P., McIlraith, S. A., Miller, T., Pearce, A. R., and Sonenberg, L. (2015). Planning over multi-agent epistemic states: A classical planning approach. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI 2015)*, pages 3327–3334. AAAI Press.
- Prezenski, S., Brechmann, A., Wolff, S., and Russwinkel, N. (2017). A cognitive modeling approach to strategy formation in dynamic decision making. *Frontiers in Psychology*, 8:1335.