SAT for Epistemic Logic Using Belief Bases

Emiliano Lorini, Benito Fabian Romero Jimenez

SAT for Epistemic Logic using Belief Bases

Emiliano Lorini[0000−0002−7014−6756] and Fabián Romero[0000−0003−2112−3061]

Institut de Recherche en Informatique de Toulouse (IRIT), Toulouse, France.
https://www.irit.fr
emiliano.lorini@irit.fr, benito.romero@irit.fr

Abstract. In [5] a new epistemic logic LDA of explicit and implicit beliefs was introduced, and in [6] we presented a tableau-based satisfability checking procedure as well as a dynamic extension for LDA. Based on such procedure, we created a portable software implementation that works for the family of multi-agent epistemic logics, as well as for the proposed dynamic extension. This software implementation runs as a library for the most common operative systems, also runs in popular IoT and robot hardware, as well as cloud environments and in server-less configurations.

1 Introduction

We believe that semantics based on explicit representation of agents' epistemic states expressed as knowledge or belief bases, are a more natural paradigm for the description of intelligent systems such as robotic and conversational agents than the Kripkean semantics commonly used for epistemic logics [3]. In order to have a tool to experiment with such semantics and explore its use, we used the logic LDA given in [6] and implemented a tableau-based satisfability procedure for it.

2 Language, Semantics and Syntax

2.1 Language of Doxastic Alternatives

The language $\mathcal{L}_{\mathsf{LDA}}$ (Language of Doxastic Alternatives) is constructed in the following way. Assume a countably infinite set of atomic propositions $Atm = \{p, q, \ldots\}$ and a finite set of agents $Agt = \{1, \ldots, n\}$.

The language $\mathcal{L}_0$, is the language of explicit beliefs defined by the grammar:

$$\alpha ::= \gamma \mid \triangle_i \alpha$$

Where γ is the grammar of classical propositional logic.

The multi-modal operator $\triangle_i \alpha$ is read as "α is a formula on agent's i belief base".

The language of implicit beliefs $\mathcal{L}_{\mathsf{LDA}}$ is defined by the grammar:

$$\phi ::= \alpha \mid \Box_i \phi \mid \Diamond_i \phi$$

Where $\alpha \in \mathcal{L}_0$ And the $\Box_i \phi$ modality can be read as "agent i can deduce ϕ from its belief base" and the modal dual $\Diamond_i \phi$ as "ϕ is consistent with agent i belief base".

The semantics for this language, is based on belief bases for a set of agents according to the following definition.

Definition 1 (Multi-agent Database). *A multi-agent database is a tuple* $B = (B_1, \ldots, B_n, S)$ *where:*
$B_i \subseteq \mathcal{L}_0, i \in Agt$, *and* $S \subseteq Atm$

Definition 2 (Satisfaction Relation).
Let $B = (B_1, \ldots, B_n, S)$ *be a multi-agent database. Then, the satisfaction relation* $\models$ *for formulas in* $\mathcal{L}_0$ *is defined as follows:*

$$\begin{aligned} B &\not\models \bot \\ B &\models p \iff p \in S \\ B &\models \neg\alpha \iff B \not\models \alpha \\ B &\models \alpha_1 \wedge \alpha_2 \iff B \models \alpha_1 \text{ and } B \models \alpha_2 \\ B &\models \triangle_i \alpha \iff \alpha \in B_i \end{aligned}$$

Definition 3 (Multi-Agent Belief Base). *A multi-agent belief base (MAB) is a multi-agent database where for all* $i \in Agt$ *and for all* $\alpha \in \mathcal{L}_0$*:*

$$\text{if } \alpha \in B_i \text{ then } B \models \alpha$$

Formulas of the language $\mathcal{L}_{\mathsf{LDA}}$ are interpreted relative to multi-agent belief bases as follows.

Definition 4 (Doxastic Alternatives). *Let* $B = (B_1, \ldots, B_n, S)$ *and* $B' = (B'_1, \ldots, B'_n, S')$ *be two multi-agent belief bases. Then,* $B\mathcal{R}_i B'$ *if and only if, for every* $\alpha \in B_i$, $B' \models \alpha$.

So, a database B' it is a doxastic alternative to the database B for an agent, if everything he could deduce in the initial database he can still deduce it in the alternative database.

Definition 5 (Satisfaction Relation (cont.)). *Let* B *a* MAB. *Then:*

$$\begin{aligned} B &\models \Box_i \varphi \iff \forall B' : \text{if } B\mathcal{R}_i B' \text{ then } B' \models \varphi \\ B &\models \Diamond_i \varphi \iff \exists B' : B\mathcal{R}_i B' \text{ and } B' \models \varphi \end{aligned}$$

Therefore, the $\Box_i$ modality, relates a database B with every database B' which is a doxastic alternative from the point of view of the agent i.

2.2 Dynamic Extension

The dynamic extension of LDA we present in our companion paper, allow us to describe actions of agents under observability conditions, this perceptive context, where the dynamic actions take place is defined by the following grammar $\mathcal{L}_{\mathsf{OBS}}$:

$$\omega ::= \mathsf{see}_{i,j} \mid \mathsf{see}_i\omega$$

The expression $\mathsf{see}_{i,j}$ Can be read as "agent i sees what agent j does". And $\mathsf{see}_i\omega$ represents the fact that "agent i sees that ω".

The language $\mathcal{L}_{\mathsf{DLDA}}$ is defined by the following grammar:

$$\chi ::= \neg\chi \mid \chi_1 \wedge \chi_2 \mid [(p, \tau, i, \Omega)]\chi \mid \phi$$

Where p is a proposition, $i \in Agt$, ϕ ranges over the language $\mathcal{L}_{\mathsf{LDA}}$,τ ranges over $\{+, -\}$ and Ω is a finite set of formulas of $\mathcal{L}_{\mathsf{OBS}}$.

The action $+p$ consists in setting the value of the atomic variable p to true, whereas the action $-p$ consists in setting the value of the atomic variable p to false. The formula $[(p, \tau, i, \Omega)]\phi$ has to be read "ϕ holds after the action τp has been performed by agent i under the perceptive context Ω".

2.3 Input Syntax

The syntax used for the library is the following. Operations and precedence order for unparenthesized expressions in $\mathcal{L}_{\mathsf{LDA}}$ are (operators are separated by commas):

$$\begin{aligned}
false &:= false, F, \bot \\
true &:= true, T, \top \\
box\ operator\ agent\ j &:= [j], \Box j \\
diamond\ operator\ agent\ j &:= < j >, \Diamond j \\
triangle\ operator\ agent\ j &:= \{j\}, \triangle j \\
negation &:= -, \sim, \neg \\
conjunction &:= \&, \wedge, /\backslash, \widehat{} \\
disjunction &:= \vee, \backslash/, | \\
implication &:= ->, \rightarrow \\
double\ implication &:= <->, \leftrightarrow \\
conjunction &:= ;
\end{aligned}$$

Propositions are strings of lowercase letters of length greater than zero, followed by zero or more digits, agents are non empty strings of digits.

We represent $\mathsf{see}_{i,j}$ in $\mathcal{L}_{\mathsf{OBS}}$ as "$i < j$" with infix right associative operator "$<$" as . We use ";" to separate observations in a perceptive context, and for the dynamic operator introduced as: $[(p, \tau, i, \Omega)]$ we will use $i + p$ or $i - p$ to represent the Boolean value of the variable p for the agent i. And "[(", ")]"

will be used to open and close the definition of the operator. For example, if $\Omega = \{\mathsf{see}_{i,i}, \mathsf{see}_{j,i}, \mathsf{see}_i\mathsf{see}_{j,i}\}$. Then the $\mathcal{L}_{\mathsf{DLDA}}$ operator $[(p, +, i, \Omega)]$ is written as:

```
[(i+p;i<i;j<i;i<j<i)]
```

For readability, we allow comments starting from a character '#' to the end of the line, and all contiguous white space characters including new lines are interpreted as a single space.

3 Tableau

Definition 6 (Tableau Rules). *A tableau rule consists of a set Γ above a line called the numerator, and a list of distinct sets $\Gamma_1, .., \Gamma_n$ separated by $|$, called the denominators:*

$$\frac{\Gamma}{\Gamma_1 \mid \ldots \mid \Gamma_n}$$

The following definition specifies the conditions under which a rule is applicable.

Definition 7 (Applicable Rule and Saturated Set). *A tableau rule is applicable to a set Γ if Γ is an instance of its numerator and Γ is not an instance of one of its denominators. We say that a set Γ is saturated if there is no rule applicable to it.*

The condition requiring that for a tableau rule to be applicable to a set Γ, Γ does not have to be an instance of one of its denominators, guarantees that when constructing a tableau we do not loop indefinitely by applying the same rule infinitely often. In the following definition, we introduce the static rules for our tableau method.

Definition 8 (Static Rules). *Let X be a finite set of formulas from $\mathcal{L}_{LDA}$, then:*

$$\bot\textit{-rule:}\ \frac{\psi; \neg\psi; X}{\bot} \qquad \wedge\textit{-rule:}\ \frac{\psi \wedge \phi; X}{\psi; \phi; \psi \wedge \phi; X}$$

$$\neg\textit{-rule:}\ \frac{\neg\neg\psi; X}{\psi; \neg\neg\psi; X} \qquad \triangle_i\textit{-rule:}\ \frac{\triangle_i\alpha; X}{\Box_i\alpha; \triangle_i\alpha; X}$$

$$\vee\textit{-rule:}\ \frac{\neg(\psi \wedge \phi); X}{\neg\psi; \neg(\psi \wedge \phi); X \mid \neg\phi; \neg(\psi \wedge \phi); X}$$

The following extra rules are used for the KD and KT variants of the logic

Definition 9 (T-rule and D-rule). *Let X be a finite set of formulas from $\mathcal{L}_{LDA}$, then:*

$$T\textit{-rule:}\ \frac{\Box_i\psi; X}{\psi; \Box_i\psi; X} \qquad D\textit{-rule:}\ \frac{\Box_i\psi; X}{\Diamond_i\psi; \Box_i\psi; X}$$

The D-rule corresponds to the property of global consistency (GC) on multi-agent belief models, while the T-rule corresponds to the property of belief correctness (BC). This correspondence is captured by the function cf such that:

$$cf(\text{D-rule}) = GC$$
$$cf(\text{T-rule}) = BC$$

Lemma 1 (Monotonicity). *For all static rule distinct to $\bot$, if Γ_k is a denominator of Γ then $\Gamma \subset \Gamma_k$*

The transitional rule allows to generate a new successor for a certain agent i.

Definition 10 (Transitional Rule). *Let X be a finite set of formulas from $\mathcal{L}_{LDA}$, then:*

$$\Diamond_i : \frac{\Diamond_i \psi; X}{\psi; \{\phi | \, \Box_i \phi \in X\}}$$

Observe that the transitional rule preserves the subsets relation, i.e. if we have two sets Γ, Δ such that $\Gamma \subseteq \Delta$ and the transtional rule applies to Γ (and therefore to Δ) then, the denominators Γ', Δ' of applying the $\Diamond_i - rule$ for an agent i to Γ and Δ respectively also have the subset relation $\Gamma' \subseteq \Delta'$.

The following definition introduces the concept of tableau.

Definition 11 (Tableau). *Let $X \subseteq \{T\text{-}rule, D\text{-}rule\}$. A tableau for Γ is a tree such that each vertex v carries a pair (Γ', ρ), where Γ' is a set of formulas and ρ is either an instance of a static rule applicable to Γ', an instance of a rule from X applicable to Γ', a transitional rule applicable to Γ' or the empty rule* nihil*, the root carries a pair (Γ, ρ) for some tableau rule ρ and for every vertex v, if v carries the pair (Γ', ρ), then the following conditions hold:*

- *if Γ' is not saturated then $\rho \neq$ nihil, and*
- *if ρ has k denominators $\Gamma_1, \ldots, \Gamma_k$ then v has exactly k children $v_1, \ldots, v_k$ such that, for every $1 \leq h \leq k$, v_h carries (Γ_h, ρ') for some tableau rule ρ',*

Observe that any sub-tree of a tableau is also a tableau.

The following definition introduces the concept of closed tableau.

Definition 12 (Closed Tableau). *A branch in a tableau is a path from the root of the tableau to an end vertex, where an end vertex is a vertex carrying a pair (Γ', nihil). A branch in a tableau is closed if its end node is of the form $(\{\bot\}, \text{nihil})$. A tableau is closed if all its branches are closed, otherwise it is open.*

From this definition and the tableau definition follows that any sub-tree of a closed tableau it is also a close tableau.

The proof of the following theorem is shown in the article

Theorem 1. *Let $\varphi \in \mathcal{L}_{LDA}$ and let $X \subseteq \{T\text{-}rule, D\text{-}rule\}$. Then, if φ is satisfiable for the class $\mathbf{M}_{\{cf(x):x \in X\}}$ then all tableaux for $\{\varphi\}$ are open.*

The algorithm that our proof induces, has to check if there is any closed-tableaux, and for that has to check every possible configuration. In order to reduce the search space, we introduce the concept of strategy.

Definition 13 (Strategy). *Let $<_\sigma$ be an order for the tableau rules. We say that a tableau follows the strategy σ if for every vertex (Γ, ρ) in the tableau, if $\rho' <_\sigma \rho$ then ρ' doesn't apply to Γ*

Theorem 2. *Let $<_\sigma$ be the following total order of the tableau rules:*

$$\bot - rule < \triangle - rule < \wedge - rule < \vee - rule < T - rule < D - rule < \Diamond - rule$$

If there is a closed tableau, then exist a closed tableau that follows the strategy σ.

To prove this we will start with the closed tableau τ and for each edge $((\Gamma, \rho), (\Gamma', \rho')) \in \tau$ if ρ' applies to Γ and $\rho' <_\sigma \rho$ then, we will create a closed tableau τ' that replace the sub-tree with root (Γ, ρ) with another closed tableau with root (Γ, ρ').

The intuition is, by using this operation repeatedly, we can "bubble sort" the tableau by "pushing upwards" the lesser operators whenever they apply to upper nodes.

We will use the following lemma in the proof.

Lemma 2 (Weakening). *Let $\Gamma, \Gamma' \subseteq \mathcal{L}_{LDA}$ If there is a closing tableau for Γ, then, there is a closing tableau for $\Gamma \cup \Gamma'$.*

Proof. If τ is a tableau for Γ then, we can create a tree τ' isomorphic to τ by mapping every node $(\Delta, \rho) \to (\Delta \cup \Gamma', \rho)$, since ρ applies to Δ then ρ applies to $\Delta \cup \Gamma'$ therefore τ' is also a tableau which applies the same rules in the same order than τ, and because τ closes, τ' also closes.

Proof Outline We will prove by each pair of rules, that if there is a closing tableau τ having an edge $((\Gamma, \rho), (\Gamma', \rho')) \in \tau$, $\rho' <_\sigma \rho$ and ρ' applies to Γ, then there is a closing tableau with (Γ, ρ') as its root.

- $(\rho, \bot)$ There is an edge $((\Gamma, \rho), (\Gamma', \bot - rule)) \in \tau$ and $\bot - rule$ applies to Γ, then, by just applying the $\bot - rule$ we have a closing tableau.
- $(\rho, \triangle_i)$ There is an edge $((\Gamma, \rho), (\Gamma', \bot - rule)) \in \tau$ and $\bot - rule$ applies to Γ, then, by just applying the $\bot - rule$ we have a closing tableau.
- $(\Diamond, \rho)$ There is an edge $(\Gamma, \Diamond - rule), (\Gamma', \rho)) \in \tau$ and ρ applies to Γ. As the rule ρ applies to Γ consider Γ_2 the result of applying ρ to Γ, as ρ is a transactional rule distinct of $\bot - rule$, by monotonocity $\Gamma' \subseteq \Gamma_2$ let Γ_2' the result of applying the $\Diamond_i - rule$ on Γ_2, as the transitional rule preserves subsets $\Gamma'' \subseteq \Gamma_2'$ as, there is a closing tableau for $(\Gamma, \Diamond - rule)$ by weakening there is a closing tableau for Γ_2.
- $(*, *)$ The rest of the cases are proven by induction on the number of rules that apply to elments of Γ, as we can assume all $\triangle_i$ rules have been applied and no applicable $\Diamond_i$ have been applied. The well know proof by invertibility of rules for tableau in modal logic [4] will work with no modifications.

4 Algorithm

We assume all formulas in B have been translated in negated normal form (NNF).

```
1: procedure IsSatisfiable(B,s)                          ▷ under the semantics s
2:     Parallel r ← SAT(B) [wait = false] do
3:         if (¬r) then
4:             return |cancel, ⊥|
5:         end if
6:     end Parallel
7:     Parallel r ← SAT(B) do
8:         if matches((ψ ∧ ¬ψ, X), B) then
9:             return |cancel, ⊥|
10:        end if
11:        if matches((△ψ; X), B) then
12:            return |current, IsSatisfiable(ψ; △ψ; □ψ; X, s)|
13:        end if
14:        if matches((ψ ∧ ξ; X), B) then
15:            return |current, IsSatisfiable(ψ; ξ; X, s)|
16:        end if
17:        if matches((ψ ∨ ξ; X), B) then
18:            return |wait, IsSatisfiable(ξ; X, s) ∨ IsSatisfiable(ψ; X, s)|
19:        end if
20:        if matches(◇ψ; X, B) then
21:            for i ← 1, n do
22:                spawn IsSatisfiable(next(i, ◇ψ; X, B), s)
23:            end for
24:        end if
25:        return |wait, ⊤|
26:    end Parallel
27: end procedure
```

5 Example

In this section, we use the logic $\mathcal{L}_{\mathsf{LDA}}$ to formalize a simple scenario of human-robot interaction in a dynamic domain inspired the famous Sally-Anne false belief's task from the psychological literature on Theory of Mind [1].

We assume that $Agt = \{h, r\}$ where h denotes the human and r denotes the robot. The scenario is depicted in Figure 1. The human and the robot are standing in front of each other on the opposite sides of a table. The robot has two boxes and two balls in front of him: box 1, box 2, a black ball and a gray ball. In the initial situation the black ball is inside box 1 and the grey ball is inside box 2. The human can perfectly observe her actions as well as the robot's actions.

Similarly, the robot can perfectly observe its actions as well as the human's actions. Moreover, the robot can see that the human can see its actions and the human can see that the robot can see her actions. Therefore, the perceptive context is described by the following set of formula from the language $\mathcal{L}_{\mathsf{OBS}}$:

$$\Omega_1 = \{\mathsf{S}_{r,r}, \mathsf{S}_{h,h}, \mathsf{S}_{r,h}, \mathsf{S}_{h,r}, \mathsf{S}_r\mathsf{S}_{h,r}, \mathsf{S}_h\mathsf{S}_{r,h}\}.$$

Let the atomic proposition *blackIn1* denote the fact that the black ball is inside box 1 and let *blackIn2* denote the fact that the black ball is inside box 2. Similarly, let *greyIn1* and *greyIn2* denote, respectively, the fact that the grey ball is inside box 1 and the fact that the grey ball is inside box 2.

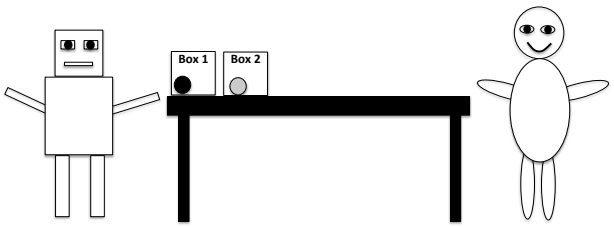

Fig. 1: Balls in the boxes scenario

We assume that in the initial situation the human does not explicitly believe that the black ball is inside box 1 and the human does not explicitly believe that the black ball is inside box 2, as she cannot see the box's content. Similarly, the human does not explicitly believe that the grey ball is inside box 1 and the human does not explicitly believe that the grey ball is inside box 2. We also assume that the robot does not explicitly believe that the human explicitly believes that the black ball is inside box 1 (resp. box 2) and that the robot does not explicitly believe that the human explicitly believes that the grey ball is inside box 1 (resp. box 2):

$$\begin{aligned} Hyp_1 \stackrel{\texttt{def}}{=} & \neg\triangle_h \mathit{blackIn1} \wedge \neg\triangle_h \mathit{blackIn2} \wedge \neg\triangle_h \mathit{greyIn1} \wedge \\ & \neg\triangle_h \mathit{greyIn2} \wedge \neg\triangle_r\triangle_h \mathit{blackIn1} \wedge \neg\triangle_r\triangle_h \mathit{blackIn2} \wedge \\ & \neg\triangle_r\triangle_h \mathit{greyIn1} \wedge \neg\triangle_r\triangle_h \mathit{greyIn2} \end{aligned}$$

Moreover, we assume that the robot explicitly believes that if the human explicitly believes that one ball is inside one box then she explicitly believes that the ball cannot be inside the other box:

$$\begin{aligned} Hyp_2 \stackrel{\texttt{def}}{=} & \triangle_r\big((\triangle_h \mathit{blackIn1} \rightarrow \triangle_h\neg \mathit{blackIn2}) \wedge \\ & (\triangle_h \mathit{blackIn2} \rightarrow \triangle_h\neg \mathit{blackIn1}) \wedge \\ & (\triangle_h \mathit{greyIn1} \rightarrow \triangle_h\neg \mathit{greyIn2}) \wedge \\ & (\triangle_h \mathit{greyIn2} \rightarrow \triangle_h\neg \mathit{greyIn1})\big) \end{aligned}$$

We can use the logic $\mathcal{L}_{\mathsf{LDA}}$ to infer that, in the perceptive context Ω_1, if the robot moves the black ball from box 1 to box 2 then, after the occurrence of the action, both the human and the robot will explicitly believe that the black ball is inside box 2, the robot will explicitly believe that the human explicitly believes that the black ball is inside box 2, and the robot will implicitly believe that the human explicitly believes that the black ball is outside box 1:

$$\begin{aligned}(\mathit{Hyp1} \wedge \mathit{Hyp2}) \rightarrow [(r, + \mathit{blackIn2}, \Omega_1)](&\triangle_r \mathit{blackIn2} \wedge \\ &\triangle_h \mathit{blackIn2} \wedge \\ &\triangle_r \triangle_h \mathit{blackIn2} \wedge \\ &\Box_r \triangle_h \neg \mathit{blackIn1})\end{aligned}$$

Now, suppose the human moves away so that she cannot see anymore what the robot does and the robot knows this. In other words, let us suppose that situation has changed into the following perceptive context Ω_2 in which the robot and the human can see their own actions but cannot see the actions of the other:

$$\Omega_2 = \{\mathsf{s}_{r,r}, \mathsf{s}_{h,h}\}.$$

In the new perceptive context Ω_2, if the robot moves the grey ball from box 2 to box 1 then, after the occurrence of the robot's action, the human will continue to believe that the black ball is inside box 2, without believing that the grey ball is inside box 1. Moreover, the robot still does not believe that the human believes that the grey ball is inside box 1:

$$\begin{aligned}(\mathit{Hyp1} \wedge \mathit{Hyp2}) \rightarrow &[(r, + \mathit{blackIn2}, \Omega_1)] \\ &[(r, + \mathit{greyIn1}, \Omega_2)](\triangle_h \mathit{blackIn2} \wedge \\ &\neg \triangle_h \mathit{greyIn1} \wedge \neg \triangle_r \triangle_h \mathit{greyIn1})\end{aligned}$$

We assume that $Agt = \{1, 2\}$ where 1 denotes a human and 2 denotes a robot. The human and the robot are standing in front of each other on the opposite sides of a table. The robot has a black ball, a grey ball, and two boxes in front of him. Initially, the human has not previous knowledge of the setting, and the robot, doesn't has any knowledge about the human's knowledge. Then, the robot puts the black ball inside the box no.2, while it is aware that the human is watching his actions. And the robot believes that the human believes that if a ball is in a given box, then that ball is not in the other box. From this setting, the robot should be able to deduce that the human believes that the black ball is not in box no.1.

```
# Observe the semicolon ';' means conjunction (with the least precendence)
#Hypotesis 1
-{1}b1 & -{1}b2;       # Human doesn't explicitly believe either ball
-{1}g1 & -{1}g2;       # is in either box
-{2}{1}b1 & -{2}{1}b2; # Robot doesn't explicitly believe the human believe
```

```
-{2}{1}g1 & -{2}{1}g2; # if either ball is in either box
#Hypotesis 2
{2}({1}b1->{1}-b2;      # Robot explicitly believes that
    {1}b2->{1}-b1;      # if human believes any ball is in either box
    {1}g1->{1}-g2;      # then it also believes that such ball is not
    {1}g1->{1}-g2);     # in the other box (here enumerated the 4 options)

# The observation context is both observing each other
# and simultaneosly aware of this fact and of themselves
-[( 2+b2; 1<1; 2<2; 1<2; 2<1; 1<2<1; 2<1<2 )](   # We set b2 true for the robot
    ({2}b2) & ({1}b2) & ({2}{1}b2);# All aware that black ball is in box 2
    [2]{1}-b1                      # Robot can conclude that human belives ...
)                                  # ... that the black ball is not in box 1
```

Which of course, after evaluating the translation, returns that is unsatisfiable. The tool is available for testing at https://tableau.irit.fr.

6 Implementation

6.1 Software, Architecture and Algorithms

We created a tool in the $F\sharp$ programming language (an open source, cross platform ML language for the Common Language Infrastructure (CLI)), that follows closely the paper as reference implementation, with the following speed improvements.

There are two separated API methods, one for the reduction of the dynamic extension, and the second for the evaluation of the satisfability given by the tableau procedure.

For the reduction of the dynamic extension, we implement the exact rewriting as specified in the paper, with no further optimization.

For the propositional case, we added a modern yet simple DPLL SAT solver, we focused more in having a clean and solid functional architecture for this rather than adding all possible heuristics, it is slower (2x-50x) than other modern SAT solver (We benchmarked against Z3 [2]), and also is much simpler (the current implementation of the SAT solver has less than 1k lines of code). However, is written in $F\sharp$, so it is exactly as portable as the library itself, which simplifies enormously the development/testing and integration as compared as using a $C++$ library which is the language most modern SAT solvers are implemented. This solver is used to discard processes, but the solution when available, is given by the tableau itself. So this is only used to help speed up execution, and it can be disabled when calling the library.

We use a reactive asynchronous execution workflow that allows us to aggressively benefit from hardware parallelism when available.

We create a process tree which is the contraction of the tableau tree on the root node and all nodes created by applying a transitional rule. Each process runs a "SAT solver" for the propositional interpretation of the set of variables, and

spawns one process for each transitional rule that would apply to the contracted tableau node. If the "SAT solver" is not satisfiable or any of the children sends a message saying it is unsatisfiable, it kills all remaining children and returns with the same message to its father. In other case, when all transitional children return a satisfactory configuration, it returns itself with the appropriate message to its father.

As we use immutable data structures, we can use shared memory between processes, in a safe and fast manner.

It is written entirely for the .net core platform, which runs in an array of architectures and operative systems, that include RaspberyPi, Linux, MacOs, Windows and the Windows 10 IoT which is rapidly increasing the array of hosts.

A trade off for the current version, is that we use a full in-memory approach. So, it runs well with models having few thousands of "modal" tableau nodes and few million propositional variables among them, but fails in much larger models, which we consider is acceptable for the kind of environments/problems the tool is designed for.

References

1. S. Baron-Cohen, A. M. Leslie, and U. Frith. Does the autistic child have a "theory of mind"? *Cognition*, 21(1):37–46, 1985.
2. Leonardo Mendonça de Moura and Nikolaj Bjørner. Z3: an efficient SMT solver. In C. R. Ramakrishnan and Jakob Rehof, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2008, Budapest, Hungary, March 29-April 6, 2008. Proceedings*, volume 4963 of *Lecture Notes in Computer Science*, pages 337–340. Springer, 2008.
3. R. Fagin and J. Y. Halpern. Belief, awareness, and limited reasoning. *Artificial Intelligence*, 34(1):39–76, 1987.
4. Rajeev Goré. *Tableau Methods for Modal and Temporal Logics*, pages 297–396. Springer Netherlands, Dordrecht, 1999.
5. E. Lorini. In praise of belief bases: Doing epistemic logic without possible worlds. In *AAAI XXXII Proceedings*, pages 1915–1922. AAAI Press, 2018.
6. Emiliano Lorini and Fabián Romero. Decision procedures for epistemic logic exploiting belief bases. In Edith Elkind, Manuela Veloso, Noa Agmon, and Matthew E. Taylor, editors, *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*, pages 944–952. International Foundation for Autonomous Agents and Multiagent Systems, 2019.