Undecidability in Epistemic Planning

Aucher, Guillaume; Bolander, Thomas

Published in:
Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence

Publication date:
2013

Document Version
Publisher's PDF, also known as Version of record

Citation (APA):
Undecidability in Epistemic Planning

Guillaume Aucher
IRISA
University of Rennes 1 - INRIA
Rennes, France
guillaume.aucher@irisa.fr

Thomas Bolander
DTU Compute
Technical University of Denmark
Copenhagen, Denmark
tobo@dtu.dk

Abstract

Dynamic epistemic logic (DEL) provides a very expressive framework for multi-agent planning that can deal with nondeterminism, partial observability, sensing actions, and arbitrary nesting of beliefs about other agents’ beliefs. However, as we show in this paper, this expressiveness comes at a price. The planning framework is undecidable, even if we allow only purely epistemic actions (actions that change only beliefs, not ontic facts). Undecidability holds already in the S5 setting with at least 2 agents, and even with 1 agent in S4. It shows that multi-agent planning is robustly undecidable if we assume that agents can reason with an arbitrary nesting of beliefs about beliefs. We also prove a corollary showing undecidability of the DEL model checking problem with the star operator on actions (iteration).

1 Introduction

Recently a number of authors have independently started developing new and very expressive frameworks for automated planning based on dynamic epistemic logic [Bolander and Andersen, 2011; Löwe et al., 2011; Aucher, 2012; Pardo and Sadrzadeh, 2012]. Dynamic epistemic logic (DEL) extends ordinary modal epistemic logic [Hintikka, 1962] by the inclusion of event models to describe actions, and a product update operator that defines how epistemic models are updated as the consequence of executing actions described through event models [Baltag et al., 1998]. Using epistemic models as states, event models as actions, and the product operator as state transition function, one immediately gets a planning formalism based on DEL.

One of the main advantages of this formalism is expressiveness. Letting states of planning tasks be epistemic models implies that we have something that generalizes belief states, the classical approach to planning with nondeterminism and partial observability [Ghallab et al., 2004].1 Compared to standard planning formalisms using belief states, the DEL-based approach has the advantage that actions (event models) encode both nondeterminism and partial observability [Andersen et al., 2012], and hence that observability can be action dependent, and we don’t need observation functions on top of action descriptions. Active sensing actions are also expressible in the DEL-based framework. Another advantage of the DEL-based framework is that it generalizes immediately to the multi-agent case. Both epistemic logic and DEL are by default multi-agent formalisms, and the single-agent situation is simply a special case. Hence the formalism provides a planning framework for multi-agent planning integrating nondeterminism and partial observability. It can be used both for adversarial and cooperative multi-agent planning. Finally, the underlying epistemic logic also allows agents to represent their beliefs about the beliefs of other agents, hence allowing them to do Theory of Mind modeling. Theory of Mind (ToM) is a concept from cognitive psychology referring to the ability of attributing mental states (beliefs, intentions, etc.) to other agents [Premack and Woodruff, 1978]. Having a ToM is essential for successful social interaction in human agents [Baron-Cohen, 1997], hence can be expected to play an equally important role in the construction of socially intelligent artificial agents.

The flip side of the expressivity advantages of the DEL-based planning framework is that the plan existence problem is undecidable in the unrestricted framework. This was proven in [Bolander and Andersen, 2011] by an encoding of Turing machines as 3-agent planning tasks (leading to a reduction of the Turing machine halting problem to the 3-agent plan existence problem). The proof made essential use of actions with postconditions, that is, ontic actions that make factual changes to the world (e.g. writing a symbol to a tape cell of a Turing machine). One could speculate that undecidability relied essentially on the inclusion of ontic actions, but in the present paper we prove this not to be the case. We prove that plan existence is undecidable even when only allowing purely epistemic (non-ontic) actions, and already for 2 agents. This is by an encoding of two-counter machines as planning tasks. We also prove that even single-agent planning is undecidable on S4 frames.

Given that we deal with multi-agent situations, it is important to specify our modeling approach, and in particular whether the modeler/planner is one of the agents. A classifi-

1 A belief state can be modeled as a set of propositional valuations, which again can be modeled as a connected S5 model of epistemic logic.
cation of the different modeling approaches and their respective formalisms can be found in [Aucher, 2010]. For ease of presentation, we follow in this article the perfect external approach of (dynamic) epistemic logic and model the situation from an external and omniscient point of view. This said, all our results in this article transfer to the other modeling approaches if we replace epistemic models with internal models or imperfect external (i.e., multi-pointed) models, which, as we said, generalize to a multi-agent setting the belief states from an external and omniscient point of view. This said, all as we said, generalize to a multi-agent setting the belief states required for the rest of the article (see [Baltag et al., 1998; van Ditmarsch et al., 2007; van Benthem, 2011] for more details). Following the DEL methodology, we split our exposition into three subsections. In Section 2.1, we recall the syntax and semantics of the epistemic language. In Section 2.2, we define epistemic models, and in Section 2.3, we define the product update. Finally, in Section 2.4, we define specific classes of epistemic and event models that will be studied in the sequel.

2 Dynamic Epistemic Logic

In this section, we present the basic notions from DEL required for the rest of the article (see [Baltag et al., 1998; van Ditmarsch et al., 2007; van Benthem, 2011] for more details). Following the DEL methodology, we split our exposition into three subsections. In Section 2.1, we recall the syntax and semantics of the epistemic language. In Section 2.2, we define epistemic models, and in Section 2.3, we define the product update. Finally, in Section 2.4, we define specific classes of epistemic and event models that will be studied in the sequel.

2.1 Epistemic Models

Throughout this article, P is a countable set of atomic propositions (propositional symbols) of cardinality at least two, and A is a non-empty finite set of agents. We will use symbols p, q, r, . . . for atomic propositions and numbers 0, 1, . . . for agents. The epistemic language L(P, A) is generated by the following BNF:

φ ::= p | ¬φ | φ ∧ φ | □iφ

where p ∈ P and i ∈ A. As usual, the intended interpretation of a formula □iφ is “agent i believes φ” or “agent i knows φ”. The formulas ◦iφ, φ ∨ ψ and φ → ψ are abbreviations of ¬□i¬φ, ¬(¬φ ∧ ¬ψ), and ¬φ ∨ ψ respectively. We define ⊤ as an abbreviation for p ∨¬p and ⊥ as an abbreviation for p ∧¬p for some arbitrarily chosen p ∈ P. The semantics of L(P, A) is defined as usual through Kripke models, here called epistemic models.

Definition 1 (Epistemic models and states). An epistemic model of L(P, A) is a triple M = (W, R, V), where W, the domain, is a finite set of worlds; R : A → 2W×W assigns an accessibility relation R(i) to each agent i ∈ A; V : P → 2W assigns a set of worlds to each atomic proposition; this is the valuation of that variable. The relation R(i) is usually abbreviated Ri, and we write v ∈ Ri(w) or wRiv when (w, v) ∈ R(i). For w ∈ W, the pair (M, w) is called an epistemic state of L(P, A).

Definition 2 (Truth conditions). Let an epistemic model M = (W, R, V) be given. Let i ∈ A, w ∈ W and φ, ψ ∈ L(P, A). Then

(M, w) |= p iff w ∈ V(p)
(M, w) |= ¬φ iff M, w ̸|= φ
(M, w) |= φ ∧ ψ iff M, w |= φ and M, w |= ψ
(M, w) |= □iφ iff for all v ∈ Ri(w), M, v |= φ

Example 1. Consider the following epistemic state of L(\{p\}; \{0, 1\}).

(M, w1) =

where 1

Each world is marked by its name followed by a list of the propositional symbols being true at the world (which is possibly empty if none holds true). Edges are labelled with the name of the relevant accessibility relations (agents). We have e.g. (M, w1) |= ¬□i0 ∧ ¬□i1¬p for i = 0, 1: neither agent knows the truth-value of p. For epistemic states (M, w) we use the symbol ⊙ to mark the designated world w.

2.2 Event Models

Dynamic Epistemic Logic (DEL) introduces the concept of event model (or action model) for modeling the changes to epistemic states brought about by the execution of actions [Baltag et al., 1998; Baltag and Moss, 2004]. Intuitively, in Definition 3 below, eQie′ means that while the possible event represented by e is occurring, agent i considers it possible that the event represented by e′ is in fact occurring.

Definition 3 (Event models and epistemic actions). An event model of L(P, A) is a triple E = (E, Q, pre), where E, the domain, is a finite non-empty set of events; Q : A → 2E×E assigns an accessibility relation Q(i) to each agent i ∈ A; pre : E → L(P, A) assigns to each event a precondition. The relation Q(i) is generally abbreviated Qi, and we write v ∈ Qi(w) or wQi,v when (w, v) ∈ Q(i). For e ∈ E, (E, e) is called an epistemic action of L(P, A).

The event e in (E, e) is intended to denote the actual event that takes place when the action is executed. Note that we assume that events do not cause factual changes in the world. Hence, we only consider so-called epistemic events and not ontic events with postconditions, as in [van Ditmarsch et al., 2005; van Benthem et al., 2006]. Our assumptions for dealing with epistemic planning will therefore also differ from the assumptions used in [Bolander and Andersen, 2011].

2.3 Product Update

Definition 4 (Applicability). An epistemic action (E, e) is applicable in an epistemic state (M, w) if (M, w) |= pre(e).

The product update yields a new epistemic state (M, w) ⊙ (E, e) representing how the new situation which was previously represented by (M, w) is perceived by the agents after the occurrence of the event represented by (E, e).

Definition 5 (Product update). Given is an epistemic action (E, e) applicable in an epistemic state (M, w), where
### 3.1 Classical Planning

Following [Ghallab et al., 2004], any classical planning domain can be represented as a restricted state-transition system $\Sigma = (S, A, \gamma)$, where $S$ is a finite or recursively enumerable set of states; $A$ is a finite set of actions; $\gamma : S \times A \rightarrow S$ is a partial and computable state-transition function. A classical planning task is then represented as a triple $(\Sigma, s_0, S_g)$, where $\Sigma$ is a restricted state-transition system; $s_0$ is the initial state; a member of $S$; $S_g$ is the set of goal states, a subset of $S$. A solution to a classical planning task $(\Sigma, s_0, S_g)$ is a finite sequence of actions (a plan) $a_1, a_2, \ldots, a_n$ such that:

1. For all $i \leq n$, $\gamma(\gamma(\ldots \gamma(\gamma(s_0, a_1), a_2), \ldots, a_{i-1}), a_i)$ is defined;

2. $\gamma(\gamma(\ldots \gamma(\gamma(s_0, a_1), a_2), \ldots, a_{n-1}), a_n) \in S_g$.

### 3.2 Epistemic Planning

**Definition 6 (Epistemic planning tasks).** An epistemic planning task is a triple $(s_0, A, \phi_g)$ where $s_0$ is a finite epistemic state, the initial state; $A$ is a finite set of finite epistemic actions; $\phi_g$ is a formula in $L(P, A)$, the goal formula.

Any epistemic planning task $(s_0, A, \phi_g)$ canonically induces a classical planning task $(S, A, \gamma, s_0, S_g)$ given by:

- $S = \{ s_0 \otimes a_1 \otimes \cdots \otimes a_n \mid n \geq 0, a_i \in A \}$.
- $S_g = \{ s \in S \mid s \models \phi_g \}$.
- $\gamma(s, a) = \begin{cases} s \otimes a & \text{if } a \text{ is applicable in } s \\ \text{undefined} & \text{otherwise} \end{cases}$.

Hence, epistemic planning tasks are special cases of classical planning tasks. A solution to an epistemic planning task $(s_0, A, \phi_g)$ is a solution to the induced classical planning task.

**Example 3.** Let $a_1$ denote the epistemic action $(E_1, e_1)$ of Example 2 and let $a_2$ denote the result of replacing 0 by 1 and 1 by 0 everywhere in $a_1$. The epistemic action $a_2$ is a private announcement of $p$ to agent 1. Now consider an epistemic planning task $(s_0, A, \phi_g)$ where $s_0 = (M, w_1)$ is the epistemic state from Example 1, and $A \supseteq \{a_1, a_2\}$. Let the goal be that both 0 and 1 know $p$, but don’t know which other knows: $\phi_g = \Box_0 p \land \Box_1 p \land \neg \Box_0 \Box_1 p \land \neg \Box_1 \Box_0 p$. It is easy to check that a solution to this epistemic planning task is the action sequence $a_1, a_2$, since we have $s_0 \otimes a_1 \otimes a_2 \models \phi_g$. Hence a solution to the task of both agents knowing $p$ without suspecting that each other does, is to first announce $p$ privately to 0 then privately to 1.

The following definition is adapted from [Erol et al., 1995].

**Definition 7 (Plan existence problem).** Let $n \geq 1$ and $L \in \{K, K4, K45, S4, S5\}$. PlanEx$(L, n)$ is the following decision problem: “Given an epistemic planning task $T = (s_0, A, \phi_g)$ where $s_0$ is an $L$-epistemic state, $A$ is a set of $L$-epistemic actions and $|A| = n$, does $T$ have a solution?”

### 4 Two-counter Machines

We will prove undecidability of the plan existence problem, PlanEx$(L, n)$, for various classes of epistemic planning tasks. Each proof is by a reduction of the halting problem for two-counter machines to the plan existence problem for the relevant class of planning tasks. So, we first introduce two-counter machines [Minsky, 1967; Hampton and Kurucz, 2012].

---

**Table:**

<table>
<thead>
<tr>
<th>$L$</th>
<th>transitive</th>
<th>Euclidean</th>
<th>reflexive</th>
</tr>
</thead>
<tbody>
<tr>
<td>K</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>KT</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>K4</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>K45</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>S4</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>S5</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 1: L-epistemic states and actions

$M = (W, R, V)$ and $E = (E, Q, pre)$. The product update of $(M, w)$ with $(E, e)$ is defined as the epistemic state $(M, w) \otimes (E, e) = ((W', R', V'), (w, e))$, where

$W' = \{(w, e) \in W \times E \mid M, w \models \text{pre}(e)\}$

$R' = \{((w, e), (v, f)) \in W' \times W' \mid wR_vu \text{ and } eQ_f\}$

$V'(p) = \{(w, e) \in W' \mid M, w \models p\}$.

**Example 2.** Continuing Example 1, the following is an example of an applicable epistemic action of $L(\{p\}, \{0, 1\})$ in $(M, w_1)$:

$$(E_1, e_1) = e_1 : p \models 0 \rightarrow 0, 1 \rightarrow 1$$

It corresponds to a private announcement of $p$ to agent 0, that is, agent 0 is told that $p$ holds (event $e_1$), but agent 1 thinks that nothing has happened (event $e_2$). The product update is calculated as follows:

$$(M, w_1) \otimes (E_1, e_1) = (w_1, e_1) : p \models 0 \rightarrow 0, 1 \rightarrow 1$$

In the updated state, agent 0 knows $p$ (since $\square p$ holds at $(w_1, e_1)$), but agent 1 didn’t learn anything (doesn’t know $p$ and believes that 0 doesn’t either).

### 2.4 Classes of Epistemic States and Actions

In this article, we consider epistemic states and actions where the accessibility relations specify specific properties, namely transitivity (for all $w, v, u, wR_vu$ and $vR_uu$ imply $wR_uu$, defined by the axiom $4$: $\Box_1 \phi \rightarrow \Box_3 \Box_1 \phi$), Euclideanness (for all $w, v, u, wR_vu$ and $wR_uu$ imply $vR_uu$, defined by the axiom $5$: $\neg \Box_1 \phi \rightarrow \Box_1 \neg \Box_1 \phi$) and reflexivity (for all $w, wR_wu$, defined by the axiom $T$: $\Box_1 \phi \rightarrow \phi$). Different conditions on the accessibility relations correspond to different assumptions on the notions of knowledge or belief [Fagin et al., 1995; Meyer and van der Hoek, 1995]. In the sequel we will refer to $L$-epistemic states and actions, where the conditions on these models are given in Figure 1.

### 3 Classical and Epistemic Planning

In this section, we briefly relate the epistemic planning approach of DEL as propounded in [Bolander and Andersen, 2011; Löwe et al., 2011] with the classical planning approach [Ghallab et al., 2004]. For more detailed connections, we refer the reader to [Bolander and Andersen, 2011].
Fig. 2: CHAIN(p, n)

Fig. 3: s(k,l,m)

Fig. 4: INC(p)
Fig. 5: DEC(p)
Fig. 6: REPL(p, n, m)

Fig. 7: The action a_M(k, l, m) when I_k = inc(0). The case I_k = inc(1) is by replacing p_2 and p_3 everywhere.

Fig. 8: The action a_M(k, l, m) when I_k = jump(j).

Definition 8 (Two-counter machines). A two-counter machine M is a finite sequence of instructions (I_0, ..., I_T), where each instruction I_t, for t < T, is from the set

{inc(i), jump(j), jzdec(i, j) | i = 0, 1, j ≤ T}

and I_T = halt. A configuration of M is a triple (k, l, m) ∈ N^3 with k being the index of the current instruction, and l, m being the current contents of counters 0 and 1, respectively. The computation function f_M : N → N^3 of M maps time steps into configurations, and is given by f_M(0) = (0, 0, 0) and if f_M(n) = (k, l, m) then

f_M(n+1) =

\[
\begin{cases}
(k + 1, l + 1, m) & \text{if } I_k = \text{inc}(0) \\
(k + 1, l, m + 1) & \text{if } I_k = \text{inc}(1) \\
(j, l, m) & \text{if } I_k = \text{jump}(j) \\
(j, l, m) & \text{if } I_k = \text{jzdec}(0, j) \text{ and } l = 0 \\
(j, l, m) & \text{if } I_k = \text{jzdec}(1, j) \text{ and } m = 0 \\
(k + 1, l - 1, m) & \text{if } I_k = \text{jzdec}(0, j) \text{ and } l > 0 \\
(k + 1, l, m - 1) & \text{if } I_k = \text{jzdec}(1, j) \text{ and } m > 0 \\
(k, l, m) & \text{if } I_k = \text{halt}
\end{cases}
\]

We say that M halts if f_M(n) = (T, l, m) for some n, l, m ∈ N.


5 Single-agent Epistemic Planning

In this section, we assume that the set A is a singleton.

5.1 The General Case

We encode the halting problem of a two-counter machine M as an epistemic planning task in three steps: 1. We define the epistemic models CHAIN(p, n) for encoding natural numbers, and the epistemic states s(k,l,m) for encoding configurations; 2. We define a finite set of epistemic actions F_M for encoding the computation function f_M; 3. We encode the halting problem as an epistemic planning task using these models.

1. Encoding of configurations. For each propositional symbol p ∈ P and each n ∈ N, we define an epistemic model CHAIN(p, n) as in Figure 2. For each (k, l, m) ∈ N^3, we define the epistemic state s(k,l,m), as in Figure 3. It encodes the configurations (k, l, m) of two-counter machines.

2. Encoding of the computation function. First, we need some formal preliminaries:

Definition 9 (Path formulas). For every n ∈ N, define γ_n := \bigvee_{i=0}^{n} \Box \bot.

Lemma 1. Let n ∈ N and let (M, w) be an epistemic state. Then (M, w) |= γ_n if there is a path of length n starting in w and ending in a world with no successor (a sink).

For each propositional symbol p ∈ P and each m, n ∈ N we define three event models INC(p), DEC(p) and REPL(p, n, m) as in Figures 4–6. We have omitted edge labels, as we are in the single-agent case.

Lemma 2. For all m, n ∈ N, for all p ∈ P,

1. CHAIN(p, n) ⊗ INC(p) = CHAIN(p, n + 1)
2. if n > 0, CHAIN(p, n) ⊗ DEC(p) = CHAIN(p, n - 1)
3. CHAIN(p, n) ⊗ REPL(p, n, m) = CHAIN(p, m).

Proof. We only prove item 1. Introducing names for the nodes and events, we can calculate as follows: CHAIN(p, n) ⊗ INC(p) =

Chain(p, n + 1).

For all k ∈ N, we define φ_k := \bigvee_{i=0}^{k} \Box (p_1 \land \neg \gamma_k \land \gamma_{k+1}). Using Lemma 1 and the definition of s(k,l,m), we immediately get that for all k, l, m, k' ∈ N:

s(k,l,m) |= φ_k' if k' = k.

(1)

Let M = (I_0, ..., I_T) be a two-counter machine. For all k < T and all l, m ∈ N, we define an epistemic action a_M(k, l, m) as in Figures 7–10 depending on the values of I_0, l and m. If k, l, m, k', l', m' ∈ N, we write (k, l, m) ≈ (k', l', m') when the following holds:

k = k' and \begin{cases} l = 0 \text{ if } l' = 0 & \text{if } I_k = \text{jzdec}(0, j) \\
0 \text{ if } m' = 0 & \text{if } I_k = \text{jzdec}(1, j)
\end{cases}

\begin{cases} m = 0 \text{ if } m' = 0 & \text{if } I_k = \text{jzdec}(1, j) \\
0 \text{ if } m' = 0 & \text{if } I_k = \text{jzdec}(0, j)
\end{cases}
\[ \phi_k \land \phi(p_2 \land \gamma_1) \]

Fig. 9: The action \( a_M(k, l, m) \) when \( I_k = jzdec(0, j), l = 0 \) or \( I_k = jzdec(1, j), m = 0 \).

\[ \phi_k \land \neg\phi(p_2 \land \gamma_1) \]

Fig. 10: The action \( a_M(k, l, m) \) when \( I_k = jzdec(0, j), l > 0 \). Case \( I_k = jzdec(1, j), m > 0 \) is by replacing \( p_2 \) and \( p_3 \).

Note that when \( (k, l, m) \approx (k', l', m') \) then \( a_M(k, l, m) = a_M(k', l', m') \), hence the following set is finite:

\[ F_M := \{ a_M(k, l, m) \mid k \in \{0, \ldots, T-1\}, l, m \in \mathbb{N} \}. \]

Lemma 3 below shows that \( F_M \) really encodes the computation steps of the computation function.

**Lemma 3.** Let \( M = (I_0, \ldots, I_T) \) be a two-counter machine, \( l, m, n \in \mathbb{N} \) and \( k < T \). Then, the following holds:

1. \( a_M(k, l, m) \) is applicable in \( s_M(n) \) if \( (k, l, m) \approx f_M(n) \).
2. \( s_{f_M(n)} \otimes a_M(f_M(n)) = s_{f_M(n+1)} \).

**Proof sketch.** Assume \( f_M(n) = (k', l', m') \). Item 1 is by case of \( I_k \). We only consider the cases \( I_k = \text{inc}(0) \) and \( I_k = \text{jump}(j) \). In these cases, \( a_M(k, l, m) \) is an epistemic action of the form \( (\mathcal{E}, e) \) with \( \text{pre}(e) = \phi_k \). Hence using equation (1) we get: \( a_M(k, l, m) \) applicable in \( s_{f_M(n)} \iff s_{(k', l', m')} \models \phi_k \iff k = k' \iff (k, l, m) \approx (k', l', m') \) because \( I_k = \text{inc}(0) \) or \( I_k = \text{jump}(j) \). Item 2 is by case of \( I_k \). We only consider the case \( I_k = \text{inc}(0) \): \( s_{f_M(n)} \otimes a_M(f_M(n)) = s_{f_M(n+1)} \).

3. **Encoding of the halting problem.** From Lemma 3, we deduce the following lemma:

**Theorem 2.** \( \text{PLANEX}(K, 1) \) is undecidable.

5.2 Epistemic Planning for K4, K45, S4 and S5

We can prove an even stronger result than Theorem 2, namely that the plan existence problem for configuration planning tasks is undecidable (for \( |P| \geq 2 \)). Due to lack of space, we cannot provide the proof of this result and we refer the interested reader to [Aucher and Bolander, 2013] for more details. The proof is similar to the proof of undecidability of the next section.

**Theorem 3.** \( \text{PLANEX}(S4, 1) \) is undecidable.

As a direct corollary of Theorem 3, we have that single-agent epistemic planning for K4 and KT are also undecidable.

Despite all these negative results, there is still room for decidability in the single-agent case if we assume that knowledge or belief are negatively introspective:

**Theorem 4.** \( \text{PLANEX}(K45, 1) \) and \( \text{PLANEX}(S5, 1) \) are decidable.

**Proof sketch.** Any formula of K45 (and hence also of S5) is provably equivalent to a normal form formula of degree 1 [Meyer and van der Hoek, 1995]. Therefore any planning task on K45 will induce a state space of finite size (up to bisimulation contraction), and hence we have decidability.

6 Multi-agent Epistemic Planning

In the multi-agent setting, we prove a strong result, namely that multi-agent epistemic planning is undecidable for any logic between K and S5. The proof of this undecidability result generalises the proof for single-agent K given in Section 5.1, but we need an extra atomic proposition \( r \) (for better readability, we use four atomic propositions \( p_1, p_2, p_3 \) and \( r \), although we could use only two). The idea underlying the proof is to replace worlds with meta-worlds, which are in fact epistemic models.

1. **Encoding of configurations.** We encode configurations as two-agent S5-epistemic states. The worlds in \( \text{CHAIN}(p, n) \) of Figure 2 are replaced with the epistemic models \( \text{METALO}(p) \) of Figure 11. The way meta-worlds are connected to each other to form a \( \text{META-CHAIN}(p, n) \) is shown in Figure 12. Then, for each configuration \( (k, l, m) \in \mathbb{N}^3 \), we define an epistemic state \( s_{(k, l, m)} \) by replacing in Figure 3 chain with \( \text{META-CHAIN} \) and by labeling the accessibility edges originating from the designated world with agent 1. Note that as we are in S5, all relations are equivalence relations, but the reflexive, symmetric and transitive closure is left implicit in figures.

2. **Encoding of the computation function.** Similarly to the case of single-agent K, we define path formulas.

**Definition 10 (Path formulas).** For all \( p \in P \) and \( n \in \mathbb{N} \), we define formulas \( \lambda_n(p) \), \( \mu_n(p) \) and \( \tau_n(p) \) inductively by:

\[
\lambda_0(p) := p \land \neg r \\
\mu_0(p) := p \land \neg r \\
\tau_0(p) := p \\
\lambda_{n+1}(p) := p \land \neg r \\
\mu_{n+1}(p) := p \land \neg r \\
\tau_{n+1}(p) := p \land \neg r 
\]

We then obtain a counterpart of Lemma 1:
Fig. 13: META-INC(p)

Fig. 14: META-DEC(p)

Fig. 15: META-REPL(p, n, m)

**Lemma 5.** For all \( p \in P \), \( n \in \mathbb{N} \), \( 0 \leq i \leq n \), \( 1 \leq j \leq 3n+3 \):

\[
\text{META-CHAIN}(p, n), w_j \models \lambda_i(p) \iff j = 3n + 3 - 3i
\]

\[
\text{META-CHAIN}(p, n), w_j \models \mu_i(p) \iff j = 3n + 2 - 3i
\]

\[
\text{META-CHAIN}(p, n), w_j \models \tau_i(p) \iff j = 3n + 1 - 3i.
\]

In other words, \( \lambda_i \) holds in the bottom world of the \((i+1)\)th to last meta-world of \( \text{META-CHAIN}(p, n) \), \( \mu_i \) in the top right world of the same meta-world and \( \tau_i \) in the top left world of the same meta-world. Now define \( \text{META-INC}(p) \), \( \text{META-DEC}(p) \) and \( \text{META-REPL}(p, n, m) \) as in Figures 13–15.

Let \( M = \langle I_0, \ldots, I_T \rangle \) be a two-counter machine. For all \( k < T \) and all \( l, m \in \mathbb{N} \), we define an epistemic action \( a_M(k, l, m) \) as in Figures 7–10 by: 1) replacing \( \text{INC} \), \( \text{DEC} \) and \( \text{REPL} \) with \( \text{META-INC} \), \( \text{META-DEC} \) and \( \text{META-REPL} \) respectively; 2) labeling the accessibility edges originating from the designated worlds with agent 1; 3) replacing \( \phi_k \) with \( \Diamond_1 \mu_k(p_1) \); 4) replacing \( \phi_k \) with \( \Diamond_2 \tau_k(p_1) \) with \( \Diamond_1 \mu_0(p_2) \).

**3. Encoding of the halting problem.** If \( \text{CHAIN}, \text{INC}, \text{DEC}, a_M \) and \( s(k, l, m) \) are replaced in Lemmata 2 and 3 with \( \text{META-CHAIN}, \text{META-INC}, \text{META-DEC}, a'_M \) and \( s'(k, l, m) \) respectively, then these Lemmata still hold. Therefore, Lemma 4 and Theorem 2 also generalize to this two-agent SS setting, and we finally obtain that:

**Theorem 5.** \( \text{PLANEX}(S5, n) \) is undecidable for any \( n \geq 2 \).

### 7 DEL Model Checking

The DEL language \( L_{DEL} \) is defined by the following BNF:

\[
\phi ::= p \mid \neg \phi \mid (\phi \land \phi) \mid (\Box_i \phi) \mid [\pi] \phi
\]

\[
\pi ::= \langle \mathcal{E}, e \rangle \mid (\pi \cup \pi) \mid (\pi; \pi) \mid \pi^*
\]

where \( p \in P \), \( i \in A \) and \( \langle \mathcal{E}, e \rangle \) is any epistemic action [van Ditmarsch et al., 2007]. In DEL, one assumes that \( |A| > 1 \). The truth conditions for the programs \( \pi \) are defined as follows:

\[
\mathcal{M}, w \models [\mathcal{E}, e] \phi \iff \mathcal{M}, w \models \text{pre}(e) \text{ implies } (\mathcal{M}, w) \otimes (\mathcal{E}, e) \models \phi
\]

\[
\mathcal{M}, w \models [\pi \cup \gamma] \phi \iff \mathcal{M}, w \models [\pi] \phi \text{ and } \mathcal{M}, w \models [\gamma] \phi
\]

\[
\mathcal{M}, w \models [\pi; \gamma] \phi \iff \mathcal{M}, w \models [\pi] \gamma \phi
\]

\[
\mathcal{M}, w \models [\pi^*] \phi \iff \text{for all finite sequences } \pi; \ldots; \pi, \mathcal{M}, w \models [\pi; \ldots; \pi] \phi
\]

The formula \( [\mathcal{E}, e] \phi \) reads as “after the execution of the epistemic action \( \langle \mathcal{E}, e \rangle \), it holds that \( \phi \)”. The model checking problem is the following: “Given an epistemic state \((\mathcal{M}, w)\), a formula \( \phi \in L_{DEL} \) is it the case that \( \mathcal{M}, w \models \phi \)?”. As an immediate corollary of our results, we have the following theorem. It complements the result of [Miller and Moss, 2005] stating that the satisfiability problem of DEL is undecidable.

**Theorem 6.** The model checking problem of the language \( L_{DEL} \) is undecidable.

**Proof.** PLANEX(S5, n) is reducible to the model checking problem of the language \( L_{DEL} \); an epistemic planning task \( T = (s_0, A, \phi_g) \) has a solution iff \( s_0 \models \neg[A^*] \neg \phi_g \) holds.

### 8 Conclusion

#### 8.1 Related Work

Alternatives to the DEL-based approach to multi-agent planning with ToM abilities can be found both in the literature on temporal epistemic logics [van der Hoek and Wooldridge, 2002] and in the literature on POMDP-based planning [Gmytrasiewicz and Doshi, 2005]. However, these alternative formalisms express planning tasks in terms of an explicitly given state space, and hence do not address how to express actions in a compact and convenient formalism (and how to possibly avoid building the entire state space when solving planning tasks). In the DEL-based formalism the state space is induced by the action descriptions as in classical planning. Note that our assumptions in DEL-based planning correspond to the infinite horizon case of planning based on POMDPs, in which already ordinary, single-agent planning is undecidable [Madani et al., 1999].

#### 8.2 Concluding Remarks

Our results are summarized in the table of Figure 16 (we recall that they hold only for \( |P| \geq 2 \)). From this table, we notice that in the single-agent setting, the property of Euclidearity (defined by Axiom 5: \( \neg \Box_i \phi \rightarrow \Box_i \neg \Box_i \phi \)) draws the borderline between decidability and undecidability: if 5 is added to K4 or S4, we immediately obtain decidability.

Given these results, an important quest of course becomes to find fragments of the formalism in which interesting problems can still be formulated, but where the complexity is comparable to the complexity of other standard planning formalisms (varying from \( \text{PSPACE} \)-completeness for classical planning [Bylander, 1994] up to \( 2^{\text{EXP}} \)-completeness for planning under nondeterminism and partial observability [Rintanen, 2004]). We leave the quest for decidable fragments to future work. Initial results in this direction can be found in [Löwe et al., 2011].
References


