DEL-based epistemic planning

Decidability and complexity

Bolander, Thomas; Charrier, Tristan; Pinchinat, Sophie; Schwarzentruber, François

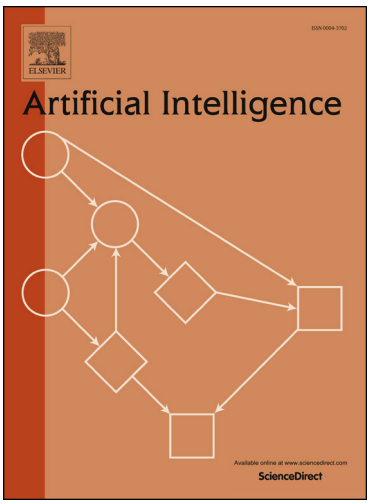
Published in:
Artificial Intelligence

Link to article, DOI:
10.1016/j.artint.2020.103304

Publication date:
2020

Document Version
Peer reviewed version

Journal Pre-proof



DEL-based Epistemic Planning: Decidability and Complexity

Thomas Bolander, Tristan Charrier, Sophie Pinchinat, Francois Schwarzentruber

PII: S0004-3702(19)30114-6
DOI: https://doi.org/10.1016/j.artint.2020.103304
Reference: ARTINT 103304

To appear in: *Artificial Intelligence*

Received date: 24 March 2019
Revised date: 11 February 2020
Accepted date: 15 May 2020

Please cite this article as: T. Bolander et al., DEL-based Epistemic Planning: Decidability and Complexity, *Artif. Intell.* (2020), 103304, doi: https://doi.org/10.1016/j.artint.2020.103304.

DEL-based Epistemic Planning: Decidability and Complexity

Thomas Bolander[a], Tristan Charrier[b], Sophie Pinchinat[b], Francois Schwarzentruber[b]

[a]*DTU Compute, Technical University of Denmark*
[b]*Univ Rennes, IRISA, France*

Abstract

Epistemic planning can be used for decision making in multi-agent systems with distributed knowledge and capabilities. Dynamic Epistemic Logic (DEL) has been shown to provide a very natural and expressive framework for epistemic planning. In this paper, we present a systematic overview of known complexity and decidability results for epistemic planning based on DEL, as well as provide some new results and improved proofs of existing results based on reductions between the problems.

Keywords: epistemic planning, dynamic epistemic logic, automated planning, decision problems, complexity theory, automata theory.

1. Introduction

In this introduction, we will first describe the transition from classical planning under full observability to multi-agent epistemic planning based on dynamic epistemic logic (DEL). We will illustrate the complications that arise when generalising from classical planning to multi-agent epistemic planning. For the sake of self-containment, we will also include a brief informal introduction to dynamic epistemic logic, whereas the formal definitions are deferred to Section 2. Readers who are both experts in automated planning and dynamic epistemic logic might consider to skip the introduction, but might be interested in a novel DEL-formalisation of the coordinated attack problem in Example 2. In this formalisation, we only use two simple message passing actions instead of relying on having infinitely many messages of unbounded modal depth. This is achieved by making use of event models with postconditions, as allowed by the general DEL setting.

Automated planning is a branch of artificial intelligence concerned with computing plans (sequences of actions) leading to some desired goal. A human or robot could e.g. have the goal of picking up a parcel at the post office, and then the problem amounts to finding a succesful sequence of actions achieving this. Epistemic planning is the enrichment of planning with epistemic notions, that is, knowledge and beliefs. The human or robot might have to reason about epistemic aspects such as: Do I know at which post office the parcel is? If not, who would be relevant to ask: Maybe the parcel is a birthday present for my daughter, and I want to ensure that she doesn't get to know, and have to plan my actions accordingly (make sure she doesn't see me with the parcel). The epistemic notions are usually formalised using an epistemic logic. Epistemic planning can naturally be seen as the combination of *automated planning* with *epistemic logic*, relying on ideas, concepts and solutions from both areas.

In general, epistemic planning considers the following problem: Given my current state of knowledge, and a desirable state of knowledge, how do I get from one to the other? It is of central importance in settings where agents need to be able to reason about their own lack of knowledge, and e.g. make plans of how to achieve the required knowledge. It is also essential in multi-agent planning, where successful coordination and collaboration can only be expected if agents are able to reason about the knowledge, uncertainty and capabilities of the other agents.

In classical STRIPS planning [23], there is only a single agent acting, actions are deterministic and the environment is fully observable. The first step towards epistemic planning is to loosen the restrictions of full observability, leading

Email addresses: tobo@dtu.dk (Thomas Bolander), tristan.charrier@gmail.com (Tristan Charrier), sophie.pinchinat@irisa.fr (Sophie Pinchinat), francois.schwarzentruber@ens-rennes.fr (Francois Schwarzentruber)

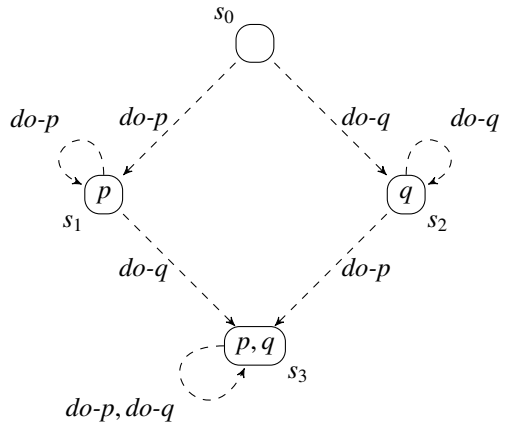


Figure 1: The state space of a simple non-epistemic planning task with initial state s_0 and goal formula $\varphi_g = p \wedge q$.

to planning under partial observability (PO planning) [13, 37]. In PO planning, partially observable states are normally represented as *belief states*: sets of fully observable states (the set of all the fully observable states that the agent consider possible, that is, cannot distinguish from the true state). Belief states are essentially the same as single-agent **S5** Kripke models in epistemic logic [25].

When moving to multi-agent environments, it becomes relevant for planning agents to reason about the mental states of other agents: Does my daughter know what I bought for her birthday? Does my husband know that I already picked up the parcel at the post office, and if not, should I inform him, so that he will not go there in vain? If a planning agent represents her own mental state as a belief state, it makes sense to also represent the mental states of other agents as belief states—or even sets of belief states, as the planning agent can (and probably most often will) have uncertainty about the exact belief states of the other agents. Multi-agent **S5** Kripke models in epistemic logic offer a very elegant formalism for representing all these belief states, including beliefs about the belief states of other agents, their beliefs about your belief states, etc. Hence, when representing states as such models, we can do planning where agents can have arbitrarily deeply nested beliefs about other agents (arbitrary levels of Theory of Mind [36]).

In the following we will refer to (pointed) multi-agent Kripke models as *epistemic states* (they might or might not be **S5** models, depending on whether we attempt to capture knowledge or e.g. belief). Planning on epistemic states gives a dramatic increase in expressive power and in applicability of the formalism to advanced multi-agent scenarios where reasoning about the mental states of other agents is crucial like the birthday parcel example above. The goal in that example could be the conjunction of three subgoals: 1) to have the parcel, 2) for the husband to know this fact, and 3) for the daughter not to know. This is clearly an epistemic goal (more precisely, the second and third subgoals are epistemic), and requires representations of the belief states of the husband and daughter. The goal can be represented in the language of *epistemic logic*, the logic underlying epistemic states. Planning to achieve the goal would amount to searching through a state space of epistemic states. Checking whether the goal holds in a specific epistemic state is then by standard model checking on such models.

The main complication and complexity of epistemic planning comes from the size and complexity of epistemic states. To illustrate this, let's first start with a basic example of single-agent non-epistemic planning under full observability.

Example 1. Figure 1 illustrates the state space of a very simple non-epistemic *planning task* over two atomic propositions p and q. In the *initial state* s_0, both p and q are false. The goal is to make both true, that is, the *goal formula* is $\varphi_g = p \wedge q$. There are two *actions* available, *do-p* that makes p true and *do-q* that makes q true (both unconditionally). Each state is represented in the figure as an oval with its set of true atomic propositions included inside the oval, and the name of the state next to it. Transitions between states are represented by dashed edges labelled by the relevant actions. Since s_3 satisfies the goal formula, and we can get to s_3 by executing *do-p* followed by *do-q* in the initial state s_0, one of the *solutions* to the planning task is the action sequence *do-p*, *do-q*.

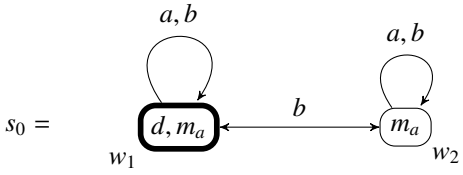


Figure 2: Example of an epistemic state $s_0 = (\mathcal{M}, w_1)$.

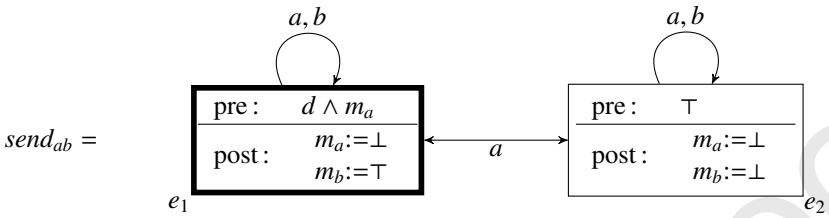


Figure 3: Example of an epistemic action $send_{ab} = (\mathcal{E}, e_1)$.

Any planning domain uses only a finite set of atomic propositions. When doing single-agent planning under full observability, it is sufficient to represent states as subsets of these propositions (the true propositions in that state). In that case, the state space is hence necessarily finite. The *plan existence problem*—deciding whether a sequence of actions achieving the goal exists—hence becomes decidable: The plan existence problem is a reachability problem on a finite graph, where we look for the reachability of a state satisfying a propositional formula.

Moving to single-agent planning under partial observability, the story is more or less the same. To account for partial observability, we use belief states (subsets of states), but since there are only finitely many states over a finite set of atomic propositions, the state space is still finite (though it can become exponentially larger than under full observability). The plan existence problem is then still a reachability problem on a finite graph, where we still look for the reachability of a state satisfying a propositional formula (in all states of the belief state). Hence decidability of the plan existence problem is still guaranteed.

When we move to epistemic states, the picture, however, changes. We can then no longer guarantee finite state spaces, and hence not guarantee decidability, as we will now informally illustrate. One of the distinguishing features of automated planning is the use of *action models* to provide compact representations of actions and their induced state transitions. Since the development of *dynamic epistemic logic* (DEL) in the late 1990s [6], such compact action descriptions have also been available to describe state transitions on epistemic states. In DEL, these action models are often called *event models*. In the following, we will refer to (pointed) event models as *epistemic actions*. The next example illustrates planning on epistemic states, using event models to describe actions.

Example 2. The *coordinated attack problem* [22] is famous in the distributed systems literature, and can be described informally as follows. Two generals, a and b, are together with their respective armies in separate camps. They need to attack their common enemy simultaneously if they want to win. However, their only way to communicate is by means of a messenger, and this messenger may be captured at any time between the two camps. Assume general a and the messenger are initially together, and that general a has decided to attack at dawn. We use the atomic proposition d to denote that 'general a will attack at dawn'. For $i = a, b$, we use the atomic proposition m_i to denote that the messenger is currently at the position of general i. Then we can use $\neg m_a \wedge \neg m_b$ to express that the messenger has been captured. The initial situation can now be described by the epistemic state of Figure 2. We call the nodes *worlds*. They correspond to the states of a belief state, that is, each node is an oval containing the atomic propositions true in the world and being labelled by its name. The world highlighted in bold, w_1, is the *actual world*: actually, general a has decided to attack at dawn (d), and actually the messenger is together with him (m_a). However, agent b doesn't know whether a has decided to attack at dawn or not, which is illustrated by the *indistinguishability edge* labelled b between w_1 and w_2: agent b also considers world w_2 possible, in which agent a has not decided to attack at dawn, since d is not true in that world.

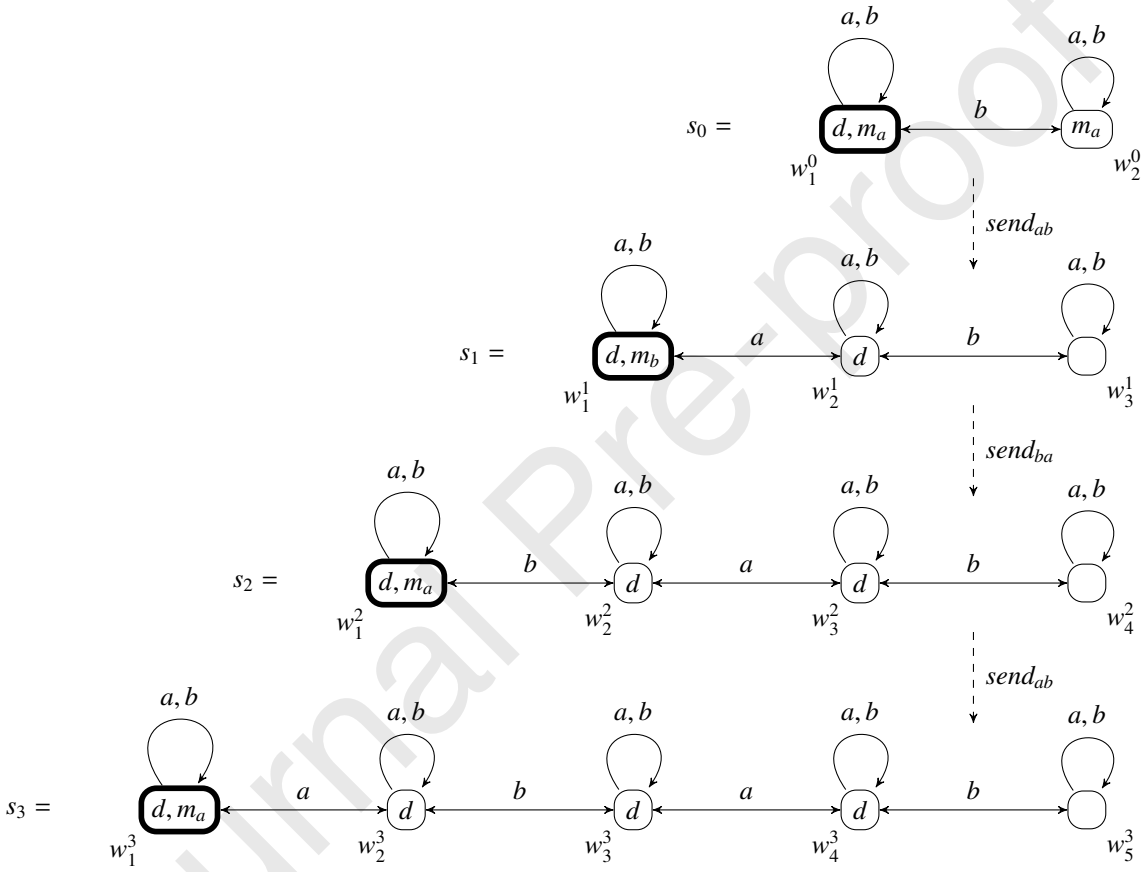


Figure 4: Initial segment of the state space of an epistemic planning task with initial state s_0 and actions $send_{ab}$ and $send_{ba}$.

The action $send_{ab}$ of sending the messenger to agent b to inform about d can be described by the epistemic action in Figure 3. Here the nodes are called *events*. They represent the possible events that can take place when the action is executed. There is a direct parallel between epistemic states and epistemic actions. Epistemic states can consist of multiple possible worlds with edges representing who cannot distinguish which worlds. Similarly, epistemic actions can consist of multiple possible events with edges representing who cannot distinguish which events. Each event is described by a *precondition* and a *postcondition* as in most action description languages for automated planning. We write "pre : $\cdots$" for preconditions and "post : $\cdots$" for postconditions. In general, in epistemic planning based on DEL, the precondition of an event can be any formula of the epistemic language. However, in the paper we will also consider epistemic planning under syntactic restrictions on the preconditions (e.g. restricting modal depth), in order to investigate how this impacts the complexity of epistemic planning. In the action $send_{ab}$, both events have propositional preconditions, that is, not using any epistemic operators. Postconditions are mappings from propositions p into formulas φ, informally represented as sets of expressions of the form $p{:=}\varphi$. The assignment $p{:=}\varphi$ expresses that the truth value of p *after* the event has occurred will be the truth value that φ had *before* the event occurred. In the action $send_{ab}$, both events only have propositional postconditions, that is, any proposition is mapped into a propositional formula (in fact, both only map into the trivial formulas $\bot$ and $\top$).

In the epistemic action of Figure 3, the *actual event*, here e_1, is highlighted in bold. Its precondition is $d \wedge m_a$ and its postcondition is $m_a{:=}\bot, m_b{:=}\top$. The postcondition makes m_a false and m_b true, which in STRIPS syntax would have been represented as $\neg m_a \wedge m_b$. This postcondition expresses that the messenger moves from general a to general b. The precondition is $d \wedge m_a$. This expresses that the event is only applicable if both d and m_a are true (if before the message is sent, general a has decided to attack at dawn and the messenger is at a). Event e_1 being the actual event means that executing the $send_{ab}$ action will actually result in event e_1 happening (the message being successfully delivered). Event e_2 represents the messenger being captured: If it occurs, the postcondition will result in the messenger being neither at a nor b (both propositions m_a and m_b are set to false after e_2 has been applied). Since general a doesn't know whether the messenger will be captured on his way to agent b, events e_1 and e_2 are indistinguishable to general a.

In addition to the action $send_{ab}$, we will assume there to be a symmetric action $send_{ba}$ where the roles of generals a and b have been swapped everywhere. The $send_{ba}$ epistemic action represents the action of successfully sending the messenger back from b to a with the message d, but this time without general b knowing whether he will arrive safely. We can think of the action sequence $send_{ab}, send_{ba}$ as first sending the message d to general b, and then general b confirming that he received the message.

Given an epistemic state s and an epistemic action α, we can compute the state resulting from executing α in s. That resulting state is denoted $s \otimes \alpha$, where the formal definition of the $\otimes$ operator is deferred to Section 2. Given the initial epistemic state s_0 of Figure 2 and the actions $send_{ab}$ and $send_{ba}$ of Figure 3, we can now as in Example 1 build a state space of all states accessible from s_0 by the available actions. Figure 4 illustrates an initial segment of this state space. An epistemic action α is only *applicable* in an epistemic state s if the precondition of the actual event of α is satisfied in the actual world of s. Hence in s_0 only $send_{ab}$ is applicable, and in any of the other states, also only one of the actions $send_{ab}$ or $send_{ba}$ is applicable. That is to be expected: After any sequence of actions, the messenger will be at one of the two generals, and can only move to the other general.

In s_0, general b does not know that d is true, since b considers the world w_2 possible. In s_1, after the messenger has delivered the message to b, general b gets to know that d is true. This is signified by general b in the actual world w_1^1 of s_1 not considering any other worlds possible. In s_1, general a however still considers it possible that b doesn't know d, since a considers it possible that b considers it possible that the actual world is w_3^1 where d is false (signified by the (a, b)-path from the actual world to w_3^1). In s_2, the message has been send from a to b and back. Now a knows that b knows d and vice versa (no (a, b)- or (b, a)-path leads to a $\neg d$-world from w_1^2). But b still considers it possible that a considers it possible that b considers it possible that $\neg d$ (due to the (b, a, b)-path to w_4^2). In general, we can see that the depth of iterated knowledge of the fact d increases with each new message being sent. However, crucially, it can also be shown that each new message passing will just increase the length of the chain model, and the rightmost world of that chain will always be a $\neg d$-world. Hence, independently of the number of messages sent, it will never become *common knowledge* that d is true, i.e., d will never be true in all worlds of an epistemic state. It can hence be shown that it can never become safe for the generals to form a coordinated attack [22]. However, if the goal of the planning task is only to achieve n-th order shared knowledge that d is true, this will be achieved after n message passings (that is, in state s_n of the state space).

What is crucial about the provided epistemic state space example is that every epistemic state has an outgoing edge leading to an epistemic state with one more world than itself. This continues indefinitely, that is, we don't have an upper bound on the size of the epistemic states of the state space. Hence state spaces can become infinite, as is exactly the case of the coordinated attack example. When state spaces are potentially infinite, decidability of the plan existence problem is no longer guaranteed. The coordinated attack example turns out to belong to a relatively simple class of epistemic planning tasks where the plan existence problem is indeed decidable, even if the state spaces of many of the planning tasks of the class are infinite. For other, more general, classes of epistemic planning tasks, we will show that the plan existence problem is undecidable. Proofs of undecidability of the plan existence problem have been done by encoding Turing machines [11], two-counter machines [2] and cellular automata [16]. The point is that certain classes of epistemic planning tasks are general enough to let the state spaces of particular planning tasks simulate the executions of such powerful types of machines/automata.

In this paper, we seek to give a roadmap of decidability and complexity results concerning the plan existence problem for different classes of epistemic planning tasks. The plan existence problem in general being undecidable for epistemic planning might make epistemic planning less attractive, but many relevant planning tasks turn out to belong to subclasses that are computationally much better behaved. To understand what makes epistemic planning hard, and understand what kind of problems in epistemic planning are less hard, it is important to delve deeper into these subclasses and the complexity of planning on them. This is exactly what this paper does. Primarily, we will look at how the complexity of pre- and post-conditions of actions affect the complexity of doing planning over those actions. If pre- and post-conditions are propositional, as for the coordinated attack problem, the plan existence problem is decidable. In addition to decidability and complexity results based on constraints on pre- and post-conditions, we also look specifically at classes of planning tasks where the epistemic states cannot "grow" (not become larger). This brings us back to planning where state spaces are guaranteed to be finite, and the plan existence problem hence decidable.

The paper is structured as follows. In Section 2, we introduce the formal machinery of dynamic epistemic logic and epistemic planning. Sections 3 and 4 are devoted to studying the complexity of the plan existence problem of epistemic planning with restrictions on the modal depth of pre- and post-conditions. First, in Section 3, we prove some novel reduction theorems that will allow us to point out the subclasses of planning tasks that are relevant to study further. Then, in Section 4, we prove decidability and undecidability for a number of these subclasses. The results of Section 4 are well-known, except Theorems 12, 14, 15 and 16 that are novel to this paper. The results of Section 4, together with the reduction theorems of Section 3, leaves us with only one open case where the decidability issue is still not settled. This open case will be discussed at the end of Section 4.

Overall, the paper contains a number of known results on (un)decidability of the plan existence problem in epistemic planning with restrictions on the modal depth of pre- and post-conditions, but also provides a new uniform presentation of them. Additionally, using these known results, it settles the (un)decidability issue of all classes of such planning tasks except one, using the novel reduction results from Section 3. The open case for which decidability is not known is approached by Theorem 12, showing that the union of the class for which decidability is not known and a simple decidable class already gives undecidability. However, the general problem is still open. The other novel results of Section 3 are some PSPACE results for variants of the plan existence problem and for certain subclasses of planning tasks not characterised by the modal depth of pre- and postconditions. The remaining two sections cover related work and a conclusion.

2. Background

In this section, we define dynamic epistemic logic. We first define the syntax of epistemic logic in Section 2.1, then define its semantics in Section 2.2 and finally introduce epistemic actions leading to dynamic epistemic logic in Section 2.3. We consider a given finite set of *agent names* (or simply *agents*) Ag and a given countable set of *atomic propositions* (or simply *atoms*) AP. Agent names are usually denoted $a, b, c, \ldots$ and atomic propositions are given names like $p, q, r, p_1, p_2, \ldots$.

2.1. Syntax of epistemic logic

Definition 1. The language of *epistemic logic* $\mathcal{L}_{\mathrm{K}}$ is defined by the following grammar.

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_a\varphi,$$

where $p \in AP$ and $a \in Ag$.

Here K_a denotes the *knowledge* (or *belief*) *modality* where $K_a\varphi$ reads "agent a knows (or believes) φ". Classically, we use $\hat{K}_a\varphi := \neg K_a \neg\varphi$, and define by the formulas $\bot = p \wedge \neg p$, $\top = \neg\bot$ and the Boolean connectives $\varphi \vee \psi = \neg(\neg\varphi \wedge \neg\psi)$, $\varphi \to \psi = \neg\varphi \vee \psi$, $\varphi \leftrightarrow \psi = (\varphi \to \psi) \wedge (\psi \to \varphi)$.

We also introduce the *modal depth* $md(\varphi)$ of a formula φ defined as: $md(p) = 0$; $md(\neg\varphi) = md(\varphi)$; $md(\varphi \wedge \psi) = \max\{md(\varphi), md(\psi)\}$; $md(K_a\varphi) = 1 + md(\varphi)$. Intuitively, $md(\varphi)$ is the maximal number of K operators in a branch of the syntax tree of φ. For instance we have $md(p \wedge q) = 0$, $md(K_a p) = 1$ and $md(K_a p \wedge K_b \hat{K}_c q) = 2$. The formulas with modal depth 0 are the *propositional formulas*.

The *size* of a formula $|\varphi|$ is defined inductively as follows: $|p| = 1$; $|\neg\varphi| = 1+|\varphi|$; $|\varphi \vee \psi| = 1+|\varphi|+|\psi|$; $|K_a\varphi| = 1+|\varphi|$.

2.2. Semantics of epistemic logic

Definition 2. A *Kripke model* is a tuple $\mathcal{M} = (\mathsf{W}, (\to_a)_{a\in Ag}, \mathsf{V})$ where the *domain* W is a non-empty finite set of *(possible) worlds*; $\to_a \subseteq \mathsf{W} \times \mathsf{W}$ is binary relation on W called the *epistemic relation* for agent a; and $\mathsf{V} : \mathsf{W} \to 2^{AP}$ assigns a *valuation* to each atomic proposition. We suppose that for every $w \in \mathsf{W}$, $\mathsf{V}(w)$ is finite. For any $w \in \mathsf{W}$, the pair $s = (\mathcal{M}, w)$ is called an *epistemic state* (or simply a *state*), and the world w is called the *designated world* or the *actual world*.

The *size* of a Kripke model $\mathcal{M} = (\mathsf{W}, (\to_a), \mathsf{V})$, written $|\mathcal{M}|$, is defined as $|\mathsf{W}| + \sum_{a\in Ag} |\to_a| + \sum_{w\in\mathsf{W}} |\mathsf{V}(w)|$. Notice that $|\mathcal{M}|$ is bounded by $|\mathsf{W}| + |Ag| \times |\mathsf{W}|^2 + |AP'| \times |\mathsf{W}|$ where $|AP'|$ is the union of the atomic propositions appearing in the image of V. The *size* of an epistemic state $s = (\mathcal{M}, w)$, written $|s|$, is the size of its underlying Kripke model $\mathcal{M}$. A Kripke model or epistemic state is called **S5** if all its epistemic relations are equivalence relations over W. On **S5**, we often call the epistemic relations *indistinguishability relations* (as we did in Section 1). We can now interpret the formulas of $\mathcal{L}_{\mathrm{K}}$ over epistemic states as follows:

$$
\begin{array}{lll}
(\mathcal{M}, w) \models p & \text{iff} & p \in \mathsf{V}(w) \\
(\mathcal{M}, w) \models \neg\varphi & \text{iff} & \mathcal{M}, w \not\models \varphi \\
(\mathcal{M}, w) \models \varphi \wedge \psi & \text{iff} & \mathcal{M}, w \models \varphi \text{ and } \mathcal{M}, w \models \psi \\
(\mathcal{M}, w) \models K_a\varphi & \text{iff} & (\mathcal{M}, w') \models \varphi \text{ for all } w' \in \mathsf{W} \text{ such that } w \to_a w'
\end{array}
$$

Example 3. We already presented an epistemic state $s_0 = (\mathcal{M}, w_1)$ in Figure 2. Each world w is represented by an oval containing the propositions of $\mathsf{V}(w)$ and labelled by its name. The edges represent the epistemic relations, and the actual world is highlighted in bold. In s_0, the atomic proposition d is true but b does not know this, which is formally expressed as $(\mathcal{M}, w_1) \models d \wedge \neg K_b d$.

2.3. Epistemic Actions and Product Update

Definition 3. An *event model* is $\mathcal{E} = (\mathsf{E}, (\to_a)_{a\in Ag}, \mathrm{pre}, \mathrm{post})$ where the *domain* E is a non-empty finite set of *events*; $\to_a \subseteq \mathsf{E} \times \mathsf{E}$ is a binary relation on E called the *epistemic relation* for agent a; $\mathrm{pre} : \mathsf{E} \to \mathcal{L}_{\mathrm{K}}$ assigns a *precondition* to each event; and $\mathrm{post} : \mathsf{E} \to (AP \to \mathcal{L}_{\mathrm{K}})$ assigns a *postcondition* to each event. We suppose that $\mathrm{post}(e)(p) \neq p$ only for a finite number of atoms p. For any $e \in \mathsf{E}$, the pair $\alpha = (\mathcal{E}, e)$ is called an *epistemic action* (or simply an *action*), and the event e is called the *designated event* or *actual event*. An event e is said to have a *propositional precondition* if $\mathrm{pre}(e)$ is a propositional formula, and is said to have a *propositional postcondition* if for every $p \in AP$, $\mathrm{post}(e)(p)$ is a propositional formula. An event model or epistemic action is called *propositional* if each of its events has propositional pre- and post-conditions.

The *size* of an event model $\mathcal{E} = (\mathsf{E}, (\to_a)_{a\in Ag}, \mathrm{pre}, \mathrm{post})$, written $|\mathcal{E}|$, is defined by

$$
|\mathcal{E}| = |\mathsf{E}| + \sum_{a\in Ag} |\to_a| + \sum_{e\in\mathsf{E}} \left(|\mathrm{pre}(e)| + \sum_{p\in AP, \mathrm{post}(e)(p)\neq p} |\mathrm{post}(e)(p)| \right)
$$

The *size* of an action $\alpha = (\mathcal{E}, e)$ is the size of its underlying event model $\mathcal{E}$. For conciseness, we specify a postcondition mapping $\mathrm{post}(e)$ by only providing the values of propositions that it modifies, namely those propositions p such that $\mathrm{post}(e)(p) \neq p$, if any; if none, we say that the action is *without postconditions* or *trivial*. By convention, if no postcondition is specified, it means it is trivial.

Example 4. We already presented an epistemic action $send_{ab} = (\mathcal{E}, e_1)$ in Figure 3. The actual event e_1 has precondition $\text{pre}(e_1) = d \wedge m_a$. The postcondition of e_1 is given by $\text{post}(e_1)(m_a) = \bot$ and $\text{post}(e_1)(m_b) = \top$, which in the figure is represented by "post : $m_a{:=}\bot, m_b{:=}\top$". It is then implicitly understood that $\text{post}(e_1)(p) = p$ for all other atomic propositions.

Note that epistemic actions are part of the domain description of planning domains, as they are the constructs in which we define the available actions, their pre- and post-conditions. The *semantics* of epistemic actions is defined via the so-called *product update* used to specify the successor state resulting from the application of an action in a state.

Definition 4. Let a state $s = (\mathcal{M}, w)$ and an action $\alpha = (\mathcal{E}, e)$ be given with $\mathcal{M} = (\mathsf{W}, (\rightarrow_a)_{a\in Ag}, \mathsf{V})$ and $\mathcal{E} = (\mathsf{E}, (\rightarrow_a)_{a\in Ag}, \text{pre}, \text{post})$. If $s \models \text{pre}(e)$ we say that α is *applicable* in s. When α is applicable in s, the *product update* of s with α is defined as

$$s \otimes \alpha = ((\mathsf{W}', (\rightarrow_a)_{a\in Ag}, \mathsf{V}'), (w, e))$$

where

- $\mathsf{W}' = \{(u, e) \in \mathsf{W} \times \mathsf{E} \mid \mathcal{M}, u \models \text{pre}(e)\}$;
- $\rightarrow_a = \{((u, e), (u', e')) \in \mathsf{W}' \times \mathsf{W}' \mid u \rightarrow_a u' \text{ and } e \rightarrow_a e'\}$;
- $\mathsf{V}'((u, e)) = \{p \in AP \mid u \models \text{post}(e)(p)\}$.

Example 5. The product update of the epistemic state $s_0 = (\mathcal{M}, w_1)$ of Figure 2 with the epistemic action $send_{ab} = (\mathcal{E}, e_1)$ of Figure 3 is the epistemic state s_1 of Figure 4. The worlds of s_1 are $w_1^1 = (w_1, e_1)$, $w_2^1 = (w_1, e_2)$ and $w_3^1 = (w_2, e_2)$. Note that there is no world (w_2, e_1) in s_1 as $(\mathcal{M}, w_2) \not\models \text{pre}(e_1)$. Note also that the submodel of s_1 achieved by removing w_1^1 is a copy of s_0 with m_a removed (made false). This is because the event e_2 has the trivial precondition $\top$, so it will apply to any world and hence generate a full copy of the original epistemic state; the postcondition of e_2 then makes both m_a and m_b false everywhere in that copy. Note that we have $s_0 \models K_a d \wedge \neg K_b d$ but $s_1 \models K_a d \wedge K_b d$, as also informally stated in Example 1. Finally, note that $send_{ba}$ is not applicable in s_0, since the precondition of the designated event e_1 of $send_{ba}$ is $d \wedge m_b$ which is not true in the designated world w_1 of s_0.

Definition 4 defines applicability of an action in a state. Sometimes we will also talk about *applicability* of a single event e in a state s, which is defined as meaning $s \models \text{pre}(e)$. Then an action α is applicable in a state s if its designated event is applicable in s.

The examples provided so far are all on **S5** models, that is, the epistemic relations are equivalence relations. Such models are often argued to be models of knowledge [25], although it might be more appropriate to call them models of *ignorance*, as the ignorance of an agent is modelled as the set of worlds that the agent cannot distinguish between. It might not be the right model for modelling belief, though, since it can not model false beliefs. Assume an agent a has a false belief that φ is true, that is, $(\mathcal{M}, w_0) \models \neg\varphi \wedge K_a\varphi$ in some model $\mathcal{M}$ with actual world w_0 (we still use the symbol K, even in cases where K more appropriately models belief). Suppose we had $(w_0, w_0) \in \rightarrow_a$. Then we wouldn't have $(\mathcal{M}, w) \models \varphi$ for all w with $(w_0, w) \in \rightarrow_a$, and hence we wouldn't have $(\mathcal{M}, w_0) \models K_a\varphi$. In other words, under the given assumptions, we cannot have $(w_0, w_0) \in \rightarrow_a$. So if a falsely believes φ, the relation $\rightarrow_a$ cannot be an equivalence relation. In other words, an agent can only have false beliefs if the epistemic relation is not an equivalence relation. Thus if we want to do epistemic planning where we instead model the beliefs of agents, and also allow agents to have false beliefs, we should not insist on the epistemic relation being an equivalence relation. In the following, we will not impose any conditions on the epistemic relation except where explicitly noted. In the general case, the way to read an a-edge from w_1 to w_2 (that is, the way to read $(w_1, w_2) \in \rightarrow_a$) is to say that *if* the actual world had been w_1, agent a would consider it possible that the actual world is w_2.

3. A classification of plan existence problems based on modal depth

We now first, in Section 3.1, define the plan existence problem. Next, in Section 3.2, we show that the plan existence problem for arbitrary modal depths of pre- and post-conditions is reducible to the plan existence problem where either pre- or post-conditions are propositional.

3.1. Plan existence problem

Definition 5. An *epistemic planning task* (or simply a *planning task*) $T = (s_0, A, \varphi_g)$ consists of an epistemic state s_0 called the *initial state*; a finite set of epistemic actions A; and a *goal formula* $\varphi_g \in \mathcal{L}_{\mathrm{K}}$. The *size* of an epistemic planning task $T = (s_0, A, \varphi_g)$ is $|T| = |s_0| + \sum_{\alpha \in A} |\alpha| + |\varphi_g|$. Let m, n be natural numbers or ∞. By $\mathcal{T}(m, n)$ we denote the class of epistemic planning tasks in which the modal depth of the event preconditions are $\leq m$, and the modal depth of the event postconditions are $\leq n$. More precisely, we have $(s_0, A, \varphi_g) \in \mathcal{T}(m, n)$ iff for every $((\mathsf{E}, (\rightarrow_a)_{a \in Ag}, \mathrm{pre}, \mathrm{post}), e) \in A$, every $e' \in \mathsf{E}$ and every $p \in AP$, $md(\mathrm{pre}(e')) \leq m$ and $md(\mathrm{post}(e')(p)) \leq n$. We furthermore use $\mathcal{T}(m, -1)$ to denote the class of planning tasks in which the modal depth of the action preconditions are $\leq m$ and the actions are without postconditions.

The planning task of the coordinated attack problem introduced in Example 2 belongs to $\mathcal{T}(0, 0)$ as all pre- and post-conditions are propositional. We say that a sequence of actions is a *solution* to an epistemic planning task if φ_g is true after applying this sequence of actions in s_0, which is formally defined as follows.

Definition 6. A *solution* to a planning task $T = (s_0, A, \varphi_g)$ is a sequence of actions $\alpha_1, \alpha_2, \ldots, \alpha_n$ from A such that for all $1 \leq i \leq n$, α_i is applicable in $s_0 \otimes \alpha_1 \otimes \cdots \otimes \alpha_{i-1}$ and

$$s_0 \otimes \alpha_1 \otimes \alpha_2 \otimes \cdots \otimes \alpha_n \models \varphi_g.$$

Example 6. We provide a variant of the consecutive numbers puzzle [45]. Consider two agents, Anne and Bill, noted a and b. Anne is given an even number and Bill an odd number, say between 0 and N, and they both know that their numbers are consecutive (for instance, Anne has 4 and Bill 3). The only allowed actions for them are to say "I don't know your number" or "I know your number". The problem is then the following: can they, only using those actions, get to know each other's numbers?

Each instance of the puzzle can be formalised as an epistemic planning task $T = (s_0, A, \varphi_g)$ as follows. The atomic propositions we use are $\{0_i, \ldots, N_i \mid i \in \{a, b\}\}$, where n_i reads "agent i has number n". Consider the instance of the puzzle having $N = 6$ and where Anne has the number 2 and Bill 1. The initial s_0 representing this instance is illustrated in Figure 5. More generally, for any instance the initial state is defined as $s_0 = (\mathcal{M}, w)$ with $\mathcal{M} = (\mathsf{W}, (\rightarrow_a)_{a \in Ag}, \mathsf{V})$ such that:

- $\mathsf{W} = \{w_{nm} \mid n, m \in \{0, \ldots, N\}, |n - m| = 1, n \text{ even}\}$;
- $w_{nm} \rightarrow_a w_{n'm'}$ if and only if $n = n'$ and $|m' - m| \in \{0, 2\}$;
- $w_{nm} \rightarrow_b w_{n'm'}$ if and only if $|n' - n| \in \{0, 2\}$ and $m = m'$;
- $\mathsf{V}(w_{nm}) = \{n_a, m_b\}$;
- $w = w_{n'm'}$ if Anne has n' and Bill has m'.

The formula "agent a knows the number of agent b" is $\varphi_a = \bigvee_{m=0}^{N} K_a m_b$. Similarly for agent b it is $\varphi_b = \bigvee_{n=0}^{N} K_b n_a$. The set of available actions of T is then $A = \{ann(\varphi) \mid \varphi \in \{\varphi_a, \neg\varphi_a, \varphi_b, \neg\varphi_b\}\}$, where the structure of the $ann(\varphi)$ is illustrated in Figure 6. The action $ann(\varphi)$ represents the *public announcement* of the formula φ [46]. The reason it is called a public announcement is that it is a singleton epistemic action that simply preserves all the worlds where φ is true. Hence, after the application of this action, φ will be universally true in the model, and become common knowledge among all agents, or, in this case, both agents. Hence it corresponds to the truth of φ being publicly broadcast to all (both) agents. See van Ditmarsch and Kooi [46] for further discussions of public announcements and common knowledge.

The final component of the planning task T is the goal formula, which we define as $\varphi_g = \varphi_a \wedge \varphi_b$. So the planning task is to achieve that both agents know the number of the other agent, and the only available actions are announcing that they know or don't know the number of the other agent. Note that this planning task belongs to $\mathcal{T}(1, -1)$, since the preconditions are of modal depth 1 and there are no postconditions. It is also in $\mathcal{T}(m, n)$ for any $m \geq 1$ and $n \geq -1$.

Figure 7 provides a solution to the planning task for the instance with initial state given by Figure 5. After the application of $ann(\neg\varphi_a)$ to the initial state, the remaining worlds are $\{w_{21}, w_{23}, w_{43}, w_{45}\}$, and after the application of $ann(\varphi_b)$ only $\{w_{21}, w_{45}\}$ remain, so the formula φ_g has become true.

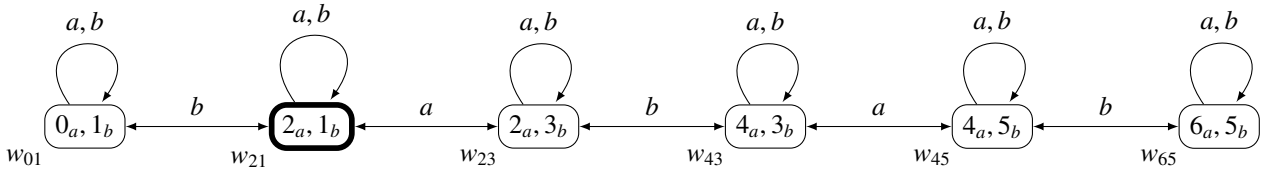


Figure 5: Initial epistemic state for the consecutive numbers puzzle with $N = 6$, and where Anne has 2 and Bill has 1.

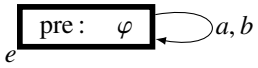


Figure 6: The epistemic action $ann(\varphi)$.

Interestingly, whether the epistemic planning task has a solution or not depends on N. Take for instance the initial state with $N = 2$ shown in Figure 8. The applicable actions are $ann(\neg\varphi_b)$ and $ann(\varphi_a)$. When applying any of those actions, the epistemic state remains unchanged, therefore it is never possible to reach a state where agent b knows her number, so φ_g can never become true.

We follow the conventions of Helmert [24] in defining the plan existence problem as follows.

Definition 7. Let $\mathcal{T}$ be a class of planning tasks. By PlanEx-$\mathcal{T}$ we denote the following decision problem, called the *plan existence problem* on $\mathcal{T}$: Given a planning task $T \in \mathcal{T}$, does T have a solution?

Now that we have defined the plan existence problem, we show that in fact, any PlanEx-$\mathcal{T}(m, n)$ problem is reducible to both PlanEx-$\mathcal{T}(0, 1)$ and PlanEx-$\mathcal{T}(1, 0)$.

3.2. Reductions to propositional pre- or postconditions

Let B and C be decision problems. We write $B \leq^P C$ when there is a polynomial-time reduction[1] from B to C, that is, a function r that transforms any instance x of B into an instance $r(x)$ of C such that: 1) x is a positive instance of B iff $r(x)$ is a positive instance of C; and 2) $r(x)$ is computable in polynomial time in the size of x. Note that if $B \leq^P C$ and C is decidable, then so is B, and, contrapositively, if B is undecidable, then so is C. We now show polynomial reductions from PlanEx-$\mathcal{T}(m, n)$ problems into other PlanEx-$\mathcal{T}(m', n')$ problems. First, since PlanEx-$\mathcal{T}(m, n)$ is a sub-problem of PlanEx-$\mathcal{T}(m + k, n + l)$ for all $k, l \geq 0$, we immediately obtain the following theorem.

Theorem 1. *For all $m \geq 0$ and $n \geq -1$, PlanEx-$\mathcal{T}(m, n) \leq^P$ PlanEx-$\mathcal{T}(m + k, n + \ell)$ for all $k, \ell \geq 0$.*

This theorem gives the diagonal reduction edges of Figure 9. We will now show how to get rid of epistemic formulas in preconditions, that is, how to compile epistemic preconditions into epistemic postconditions. Furthermore, we will be able to make sure that all epistemic postconditions are of modal depth at most 1. In other words, we will show how any planning task in $\mathcal{T}(m, n)$ can be turned into an equivalent planning task in $\mathcal{T}(0, 1)$. We first illustrate the construction on a concrete epistemic action, the action α of Figure 10. For each formula φ in α (that may appear as a precondition or in a postcondition assignment in α), for each epistemic subformula ψ of φ, we introduce a fresh atomic proposition p_ψ. In our example, we generate the propositions $p_{K_a p}$, $p_{K_b p}$, $p_{K_b q}$ and $p_{K_a K_b p}$.

Action α is then simulated by the actions $\alpha_{choose}, \alpha_1, \alpha_2, \alpha'$, executed in this order. Action α_{choose} assigns a certain proposition p_α to $\top$, to impose that the actions $\alpha_1, \alpha_2, \alpha'$ are executed just after α_{choose}. Then, action α_1 stores the values of epistemic formulas of modal depth 1 into their associated propositions ($p_{K_a p}, p_{K_b p}, p_{K_b q}$ in the example). Action α_2 does the same for epistemic formulas of modal depth 2 ($p_{K_a K_b p}$ in the example). Finally, action α' is a copy of α where each epistemic formula φ has been replaced by its corresponding proposition p_φ, and the new propositions are reset to false. The order of execution is guaranteed by the introduction of new propositions p_1, p_2 and p_{exec}. The new goal formula φ'_g becomes $\varphi_g \wedge \bigwedge_{\alpha \in A} \neg p_\alpha$, where φ_g is the original goal formula and the $\bigwedge_{\alpha \in A} \neg p_\alpha$ part ensures the evaluation of φ_g just after some α'. This idea is generalized by the following theorem.

[1] The reader may refer to Padadimitriou [35].

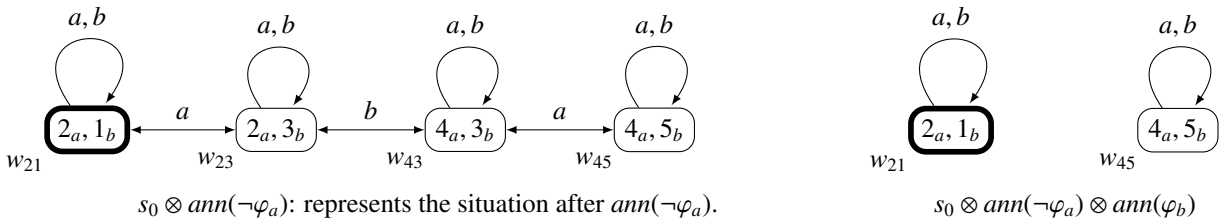


Figure 7: Illustration of the solution to the consecutive number puzzle with initial state given by Figure 5.

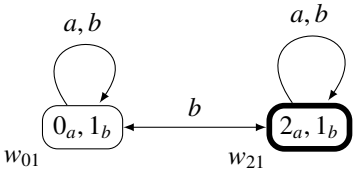


Figure 8: Initial epistemic state for the consecutive numbers puzzle with $N = 2$, where Anne has 2 and Bill has 1.

Theorem 2. *For all $m \geq 0$ and $n \geq -1$, PlanEx-$\mathcal{T}(m,n) \leq^P$ PlanEx-$\mathcal{T}(0,1)$.*

Proof. The proof is by generalising the construction above, and is provided in Appendix A. □

This theorem gives the horizontal reduction edge of Figure 9, the reduction from PlanEx-$T(1,0)$ to PlanEx-$T(0,1)$. Instead of getting rid of epistemic formulas in preconditions, we can choose to get rid of them in postconditions, that is, to compile epistemic postconditions into epistemic preconditions. And, again, we can make sure that all epistemic conditions are of modal depth at most 1. In other words, we can prove that any planning task in $\mathcal{T}(m,n)$ can be turned into an equivalent planning task in $\mathcal{T}(1,0)$, that is, PlanEx-$\mathcal{T}(m,n) \leq^P$ PlanEx-$\mathcal{T}(1,0)$. We prove it in the following way. First we show that PlanEx-$\mathcal{T}(m,n) \leq^P$ PlanEx-$\mathcal{T}(\max\{m,n\},0)$. From this we can conclude PlanEx-$\mathcal{T}(0,1) \leq^P$ PlanEx-$\mathcal{T}(1,0)$ (letting $m = 0$ and $n = 1$). Using Theorem 2, we can then finally conclude PlanEx-$\mathcal{T}(m,n) \leq^P$ PlanEx-$\mathcal{T}(1,0)$ (by transitivity of $\leq^P$). To prove PlanEx-$\mathcal{T}(m,n) \leq^P$ PlanEx-$\mathcal{T}(\max\{m,n\},0)$, we simulate the execution of any action α with the execution of two actions, α_{assign} and α' with propositional postconditions. We add two types of atomic propositions: $p_{\text{post}(e)(p)}$ that stores the value of $\text{post}(e)(p)$ and p_α that is true if we are currently executing α_{assign} and α'. The action α_{assign} stores the truth value of $\text{post}(e)(p)$ in the proposition $p_{\text{post}(e)(p)}$ by checking its value in the precondition. Then α' is a copy of α where each $\text{post}(e)(p)$ is now replaced by $p_{\text{post}(e)(p)}$. The order of execution α_{assign} then α' is ensured by p_α.

Theorem 3. *For all $m \geq 0$ and $n \geq -1$, PlanEx-$\mathcal{T}(m,n) \leq^P$ PlanEx-$\mathcal{T}(1,0)$.*

Proof. The proof follows the idea sketched above, and is provided in Appendix B. □

4. Complexity results for the plan existence problem based on modal depth

We now detail the results on the complexity of PlanEx-$\mathcal{T}(m,n)$ depending on m and n. In Section 4.1.1, we prove the decidability of PlanEx-$\mathcal{T}(0,0)$, that is, that planning on propositional pre- and post-conditions is decidable. In Section 4.1.2, we show that the subclass $\mathcal{T}(0,-1)$ of $\mathcal{T}(0,0)$ is PSPACE-complete. That is, planning with propositional preconditions and no postconditions is PSPACE-complete. Then, in Section 4.2.1, we explain the proof of undecidability for PlanEx-$\mathcal{T}(1,0)$, which allows us to deduce undecidability for all other PlanEx-$\mathcal{T}(m,n)$ with $m \geq 1$ and $n \geq 0$ using Theorem 1. Using Theorem 2, it allows us to furthermore deduce undecidability of PlanEx-$\mathcal{T}(m,n)$ for all $m \geq 0$ and $n \geq 1$, see Figure 9. In other words, the plan existence problem is undecidable on any class of planning tasks that include all tasks with preconditions of modal depth ≤ 1 and with propositional postconditions; as well

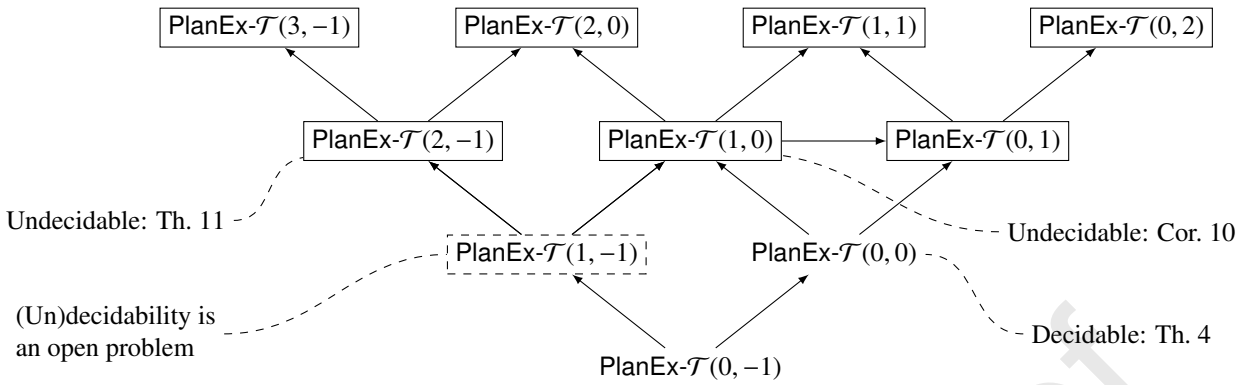


Figure 9: A lattice of polynomial-time reductions between plan existence problem of the form PlanEx-$\mathcal{T}(m,n)$. An edge from A to B means $A \leq^P B$, so the problems get increasingly harder the higher we move up in the lattice. All the diagonal edges are by Theorem 1, whereas the horizontal edge is by Theorem 2. Undecidable problems have a solid rectangle as border. Our main undecidability results are marked with theorem numbers, and undecidability of all the other problems with a solid border follows using the reduction edges. Decidable problems have no border. Our main decidability result is marked with its theorem number. The decidability of the other problem with no border follows using the reduction edges. The only problem for which decidability is still an open problem (after having proved all theorems of this paper) is marked with a dashed border.

as on any class of planning tasks that include all tasks with propositional preconditions and postconditions of modal depth ≤ 1. Afterwards, in Section 4.2.2, we explain the proof of undecidability for PlanEx-$\mathcal{T}(2,-1)$, showing that if we allow preconditions of modal depth 2, then even without postconditions, the plan existence problem is undecidable. Given these results, the only PlanEx-$\mathcal{T}(m,n)$ for which (un)decidability is still not decided is PlanEx-$\mathcal{T}(1,-1)$, as is easily seen from Figure 9. In Section 4.3, we discuss progress on the open case of PlanEx-$\mathcal{T}(1,-1)$ and prove that we can reduce PlanEx-$\mathcal{T}(m,n)$ into PlanEx-$(\mathcal{T}(1,-1) \cup \mathcal{T}(0,0))$, hence proving PlanEx-$(\mathcal{T}(1,-1) \cup \mathcal{T}(0,0))$ to be undecidable.

4.1. Decidable cases

4.1.1. The case of propositional pre- and post-conditions

The class of planning tasks with propositional pre- and post-conditions is $\mathcal{T}(0,0)$. We already gave examples of such tasks: Any goal formula for the coordinated attack problem considered in Example 2 will give a planning task in $\mathcal{T}(0,0)$. As illustrated in that example, the state space of the coordinated attack problem is infinite. However, the plan existence problem for planning tasks in $\mathcal{T}(0,0)$ is still decidable, as we will now show.

Theorem 4. *PlanEx-$\mathcal{T}(0,0)$ is decidable.*

The rest of this subsection is dedicated to a proof of the theorem. Although there already exists an ad-hoc proof of this result by Yu et al. [49], we provide here the proof from Aucher et al. [3]. This proof resorts to a powerful result in classical logic, namely the decidability of first-order logic (FO) on *automatic structures*, that we will explain beforehand. Indeed, first, one can show that any initial state and set of actions of a planning task in $\mathcal{T}(0,0)$ yields an infinite first-order structure, that turns out to be an automatic structure called a *DEL structure*, and whose elements are sequences of applicable events. Second, since one can translate any epistemic goal formula into FO, the existence of a solution can be rephrased as an FO query on the DEL structure, and is therefore decidable. Moreover, by taking advantage of automata constructions underlying the model-checking procedure of automatic structures against FO properties, we effectively build a finite-state automaton whose language is the set of all solutions to the planning task.

We first recall the seminal result that FO logic is decidable on *automatic structures*, by defining these structures and recalling FO, and by explaining the model-checking procedure. The interested reader may refer to Blumensath and Grädel [10] and Rubin [38] for further details.

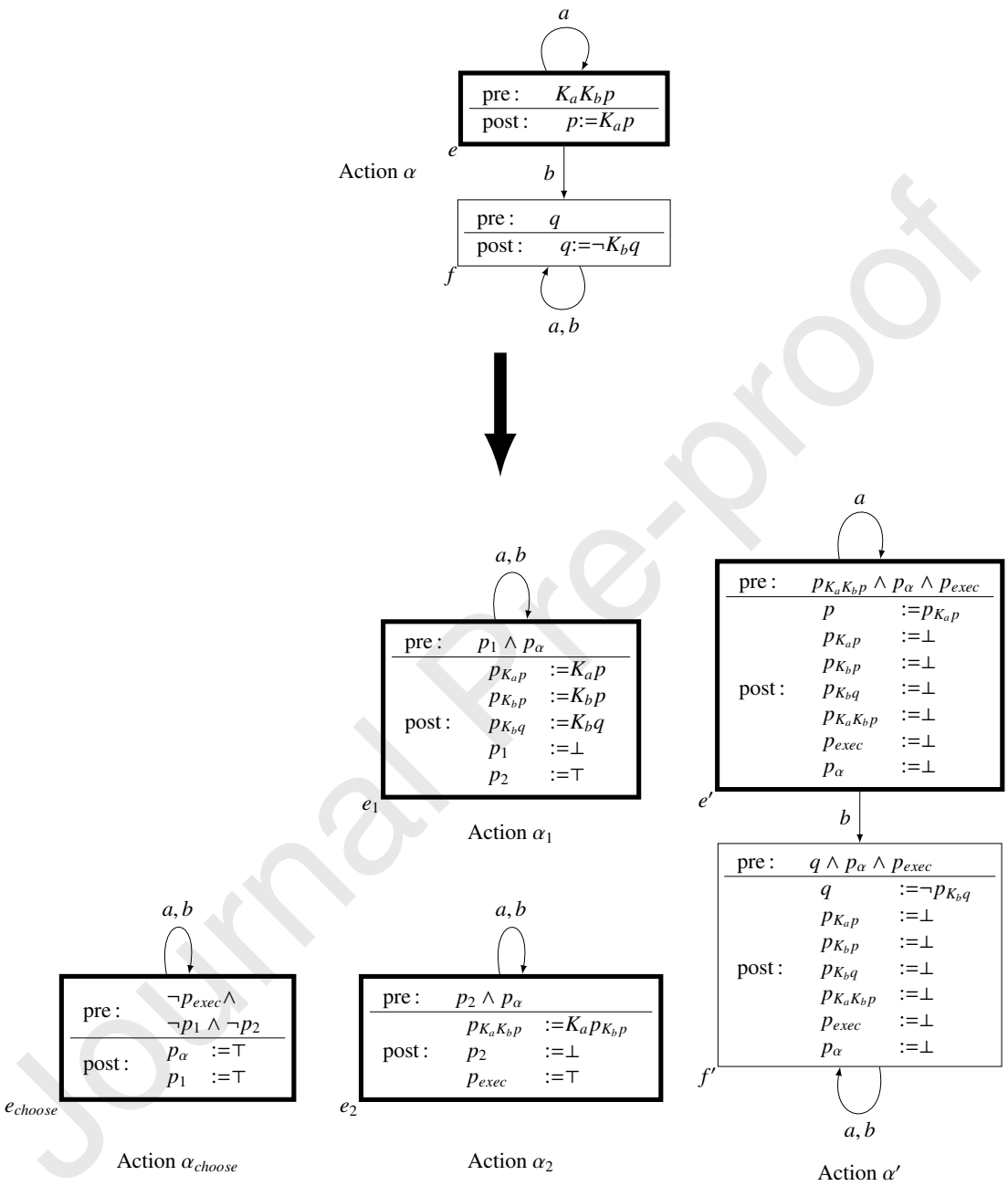


Figure 10: Example of actions $\alpha_{choose}, \alpha_1, \alpha_2, \alpha'$ for the $\mathcal{T}(0,1)$ construction.

Structures and logics. First-order logic is interpreted over relational structures.

Definition 8. A *relational structure* is a structure of the form $\mathcal{S} = \langle D, R_1 \dots R_p \rangle$ where:

- D is a non-empty set called the *domain*;
- $R_1 \dots R_p$ are *relations* over D of arity $r_1 \dots r_p$, respectively (i.e. $R_i \subseteq D^{r_i}$).

We take the convention to write $R_i(d_1, \dots, d_{r_i})$ for $(d_1, \dots, d_{r_i}) \in R_i$.

Example 7. The structure $\mathcal{T}_2 = \langle \{1,2\}^*, S_1, S_2 \rangle$ is the 2-ary tree, called the *infinite full binary tree*, that is the relational structure whose domain is the set of node addresses and whose relations are the two binary relations S_1 and S_2 relating a node with its first child and second child, respectively. The structure $\mathcal{T}_2$ can be extended as the structure $\mathcal{T}_2^{\mathsf{el}} = \langle \{1,2\}^*, S_1, S_2, \mathsf{el} \rangle$ with the additional binary relation "at equal level in the tree", namely $\mathsf{el}(x, y)$ holds if, and only if, $|x| = |y|$.

The set of symbols $\{R_1 \dots R_p\}$ is called the *signature* of $\mathcal{S} = \langle D, R_1 \dots R_p \rangle$. We will loosely use symbol relations $R_1, \dots, R_p$ to denote the predicates of the *first-order logic (FO)* interpreted over the relational structure $\mathcal{S}$, which we define now. The formulas of FO (over signature $\{R_1 \dots R_p\}$) conform to the following syntax:

$$\Psi ::= R_i(x_1 \dots x_{r_i}) \mid \neg\Psi \mid (\Psi \wedge \Psi) \mid \exists x \Psi$$

where $x, x_1, \dots, x_{r_i}$ are first-order variables whose assignment ranges over the domain D of relational structures.

We write $\mathcal{S}, [x_i \mapsto d_i]_{1 \le i \le n} \models \Psi[x_1 \dots x_n]$ to express that formula $\Psi(x_1 \dots x_n)$ with free variables $x_1, \dots, x_n$ is true in $\mathcal{S}$ whenever $x_1, \dots, x_n$ are assigned to $d_1, \dots, d_n$ respectively, and we denote by $\Psi^{\mathcal{S}}$ the n-ary relation made of all the tuples that "satisfy" Ψ:

$$\Psi^{\mathcal{S}} = \{(d_1 \dots d_n) \in D^n \mid \mathcal{S}, [x_i \mapsto d_i]_{1 \le i \le n} \models \Psi[x_1 \dots x_n]\}$$

Automatic presentations. We now describe how some relational structures can be encoded using formal languages, following the presentation of Rubin [38]. By *alphabet* we mean a finite set of symbols, named *letters*. A *word* over Σ is a finite sequence of letters. We denote by Σ^* the set of such sequences. For $u \in \Sigma^*$, we write $|u|$ for its length, and for any $0 \le n \ge |u|$, we let $u[n]$ be the $n+1$-th letter of u); $u[0]$ is the first letter of word u. We assume familiarity with the basic definitions of automata theory and the properties of regular languages.

As we will see, the domain of a relational structure will be encoded as a regular language over some alphabet Σ. A relation of a relational structure is then a set of tuples of words over Σ, and we therefore need to decide on a convention for how to represent those tuples. Given an alphabet Σ, we let $\square$ be a fresh *padding* symbol, and we let $\Sigma_\square$ be $\Sigma \uplus \{\square\}$. Now, k-tuples of words will be encoded as words on the product alphabet $\Sigma_\square \times \Sigma_\square \times \dots \Sigma_\square$ (k-times), where the padding symbol $\square$ is used to align the words of the tuple, as they may have different lengths. For example, say with $\Sigma = \{\mathsf{a}, \mathsf{b}\}$, we would align the three words aaba, b, and ba as the four-letter word $\binom{\mathsf{a}}{\mathsf{b}}\binom{\mathsf{a}}{\square}\binom{\mathsf{b}}{\square}\binom{\mathsf{a}}{\square}$ over the product alphabet $\Sigma_\square \times \Sigma_\square \times \Sigma_\square$ (the elements of which are triples that we write vertically). This notion of alignment is formally captured as the *convolution* of words, defined as follows.

Definition 9. The *convolution* $u \otimes v$ of two words $u, v \in \Sigma^*$ is a word in $(\Sigma_\square \times \Sigma_\square)^*$ of length $\max(|u|, |v|)$ defined by

$$(u \otimes v)[i] = \begin{cases} \binom{u[i]}{v[i]} & \text{if } i < \min(|u|, |v|), \text{ that is, if both } u[i] \text{ and } v[i] \text{ are letters of } \Sigma, \\ \binom{u[i]}{\square} & \text{if } |v| \le i < |u|, \\ \binom{\square}{v[i]} & \text{if } |u| \le i < |v|. \end{cases}$$

Notice that according to Definition 9, word $u \otimes v$ does not carry the product letter $\binom{\square}{\square}$. The convolution is defined similarly for k-tuples of words.

We now turn to the definition of automatic structures that rely on the notion of automatic presentations. Basically, a relational structure has an automatic presentation if (i) its domain is encoded as some regular language, and (ii) each of its relations, seen as the set of word convolutions that encode the elements of the relation, is also a regular language.

Definition 10. An *automatic presentation* of a relational structure $\mathcal{S} = \langle D, R_1 \dots R_p \rangle$ is a tuple $(\mathcal{A}_D, \mathcal{A}_1, \dots, \mathcal{A}_p)$ of finite-state automata such that:

(1) There exists a one-to-one *encoding function* $\mathtt{enc} : D \to L(\mathcal{A}_D)$. The inverse $\mathtt{enc}^{-1}$ of the encoding function is the *decoding function* that retrieves the original element of D from its encoding in $L(\mathcal{A}_D)$.

For an arbitrary relation $R \subseteq D^r$, we let $\mathtt{enc}(R) := \{\mathtt{enc}(d_1) \otimes \dots \otimes \mathtt{enc}(d_r) \,|\, R(d_1, \dots, d_r)\}$.

(2) Every relation R_i (say of arity r_i) among $R_1, \dots, R_p$ is captured by the language of automaton $\mathcal{A}_i$ that encodes the r_i-tuples in R_i: $L(\mathcal{A}_i) = \mathtt{enc}(R_i)$.

A structure is *automatic* if it has an automatic presentation.

Remark 1. We may assume that equality is among the relations R_i, represented by the regular language $\{u \otimes u \,|\, u \in L_D\}$. In the literature, the standard definition of automatic presentations allows an element to have several encodings, whenever equality can be presented by some regular language. However, both definitions yield the same class of structures [10].

Example 8. Finite structures are automatic, since finite languages are regular. Both trees $\mathcal{T}_2$ and $\mathcal{T}_2^{\mathsf{el}}$ of Example 7 are also automatic. Finally, so is $\langle \mathbb{N}, \leq \rangle$ with a natural automatic presentation over a unary alphabet $\{1\}$ by letting $\mathtt{enc}(n)$ be the unary representation of n, that is, the word 1^n. Automaton $\mathcal{A}_{\leq}$ depicted in Figure 11 verifies that, provided an input as the convolution of two finite sequences of 1's, there are less 1's in the former than in the latter.

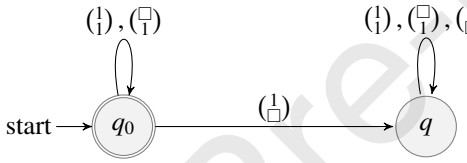


Figure 11: The finite-state automaton $\mathcal{A}_{\leq}$ of Example 8 (ii).

For example, consider the pair of natural numbers $(2, 3)$, that is encoded by the convolution $11 \otimes 111 = \binom{1}{1}\binom{1}{1}\binom{\square}{1}$ and feed automaton $\mathcal{A}_{\leq}$ with it. The automaton execution for this input will be

$$q_0 \xrightarrow{\binom{1}{1}} q_0 \xrightarrow{\binom{1}{1}} q_0 \xrightarrow{\binom{\square}{1}} q_0,$$

which is accepting, since state q_0 is an accepting state. This is what we expect since it is indeed the case that $(2, 3) \in \leq$. On the contrary, feeding this automaton with the word $11 \otimes \varepsilon$, that encodes the pair $(2, 0)$, yields execution

$$q_0 \xrightarrow{\binom{q}{\square}} q \xrightarrow{\binom{q}{\square}} q,$$

which is rejecting since state q is not accepting. This is of course also as expected, since $(2, 0) \notin \leq$. Note that the automaton would also reject malformed input by mere blocking.

We recall the known fundamental theorem regarding the model checking of automatic structures against FO, that we first define.

Definition 11. FO Model checking

Input : An automatic presentation of some structure $\mathcal{S}$, and a closed FO-formula Ψ.
Output : *Yes* if $\mathcal{S} \models \Psi$, *No* otherwise.

The following theorem expresses that the verification of FO-definable properties over automatic structures with an effective presentation can be automated, see for example [38, Th. 3.1].

Theorem 5. *FO model checking is decidable.*

The main ingredient of the proof of Theorem 5 relies on the fact that

$$\Psi^{\mathcal{S}} = \{(d_1 \dots d_n) \in D^n \mid \mathcal{S}, [x_i \mapsto d_i]_{1 \leq i \leq n} \models \Psi[x_1 \dots x_n]\}$$

is a regular n-ary relation, with an effective construction of a finite automaton that recognizes it.

Proposition 6. *There is an algorithm that, given an automatic presentation of a relational structure $\mathcal{S}$ with encoding function* enc*, and a formula $\Psi(x_1, \dots, x_n)$ in FO, constructs an automaton $\mathcal{A}_\Psi^{\texttt{enc}}$ that recognizes* $\texttt{enc}(\Psi^{\mathcal{S}})$.

The proof of Proposition 6 relies on the closure of regular languages under intersection, complementation, and projection over components; these operations faithfully reflect the logical conjunction, negation, and existential quantification, respectively [38]. When enc is clear from the context or not necessary to mention explicitly, we will often write $\mathcal{A}_\Psi$ for $\mathcal{A}_\Psi^{\texttt{enc}}$.

In the following, we describe the relational structures arising from epistemic planning tasks, and show that if all actions are propositional, they are automatic for the identity encoding function.

DEL structures. Let an epistemic planning task $T = (s_0, A, \varphi_g)$ be given where every action is propositional. We can turn the epistemic actions of A into a single epistemic action α, by taking the disjoint union of the individual elements of A. This will give an action with multiple designated events, something we did not consider yet in this paper, although fairly standard in many presentations of epistemic planning [11].

However, unlike most often the case in the literature, we do not interpret actions with multiple designated events as a way to represent non-determinism (nature non-deterministically choses which designated event will occur). Here, the multiple designated events feature is used to succinctly represent the choice of the planning agent between the original epistemic actions in A: if we have $A = \{\alpha_1, \alpha_2\}$ with $\alpha_1 = (\mathcal{E}_1, e_1)$ and $\alpha_2 = (\mathcal{E}_2, e_2)$, and we let α be the disjoint union of α_1 and α_2, then the choice of the agent between executing α_1 and α_2 amounts to choosing between the two designated events e_1 and e_2 in α.

We therefore without loss of generality consider planning tasks of the form $T = (s_0, \alpha, \varphi_g)$ hence containing a single action α written $\alpha = (\mathcal{E}, \mathsf{E_o})$ where $\mathsf{E_o} \subseteq \mathsf{E}$ represents the choices of the planning agent. The number of standard actions represented by such a multi-pointed action is $|\mathsf{E_o}|$. We extend the definition of product update to states and actions with multiple designated events in the obvious way: A world (w, e) of $s \otimes \alpha$ is *designated* if w is a designated world of s and e is a designated event of α.

Given a planning task of form $T = (s_0, \alpha, \varphi_g)$, we call the pair consisting of the first two elements (s_0, α) a *DEL presentation*. A DEL presentation naturally induces a state space of the planning task, cf. Section 1. This state space can be seen as a relational structure that we will call a *DEL structure*, to be defined next.

Following Maubert [32], given a DEL presentation (s_0, α), we incorporate the initial state $s_0 = (\mathcal{M}, w_0)$ and the infinitely many product updates $s_0 \otimes \alpha^n$ $(n \in \mathbb{N})$[2] into a single relational structure, called a *DEL structure*, denoted by $s_0\alpha^*$. The domain of $s_0\alpha^*$ is written $\mathcal{H}$ and is composed of all worlds of all updates $s_0 \otimes \alpha^n$ $(n \in \mathbb{N})$ called *histories*: they are of the form $h = we_1 \dots e_n$ where w is a world of $\mathcal{M}$ and $e_1, \dots, e_n$ are events of α, e.g. the world $((w, e_1), e_2)$ is denoted we_1e_2. Notice that because $we_1 \dots e_n$ belongs to $s_0 \otimes \alpha^n$, each event e_i is applicable in the state $s_0 \otimes \alpha^{i-1}$. We take the convention to write $\mathsf{V}(h)$ for the set of atomic propositions that hold after history h has taken place. Among the histories of $\mathcal{H}$, we distinguish those that start in the designated world w_0 of s_0 and are followed by designated events in $\mathsf{E_o}$. Such histories are called *designated histories*. More formally, DEL structures are defined as follows.

[2] We use $s_0 \otimes \alpha^n$ as a shorthand for $s_0 \otimes \alpha \otimes \alpha \otimes \dots \otimes \alpha$ with n occurrences of α. It is also possible to define product updates between actions, so that α^n will be an action representing n iterations of α, though we will not do that here.

Definition 12 ([32, Def. 54, p. 105])**.** Let $s_0 = (\mathcal{M}, w_0)$ be a state and let $\alpha = (\mathcal{E}, \mathsf{E}_{\mathbf{o}})$ be an action where $\mathcal{E} = (\mathsf{E}, (\rightarrow_a)_{a\in Ag}, \mathrm{pre}, \mathrm{post})$. The *DEL structure* induced by the DEL presentation (s_0, α) is the relational structure $s_0\alpha^* = (\mathcal{H}, (S_e)_{e\in\mathsf{E}}, (\rightarrow_a)_{a\in Ag}, (p)_{p\in AP}, \mathbf{o})$, where all relations S_e and $\rightarrow_a$ are binary relations and relations p and $\mathbf{o}$ are unary, and where:

- the domain $\mathcal{H}$ is the disjoint union of the sets of worlds of $s_0 \otimes \alpha^n$ – namely, the *histories*;
- for each $e \in \mathsf{E}$, $S_e(h, h')$ whenever $h' = he$;
- for each agent $a \in Ag$, $\rightarrow_a (h, h')$ whenever h and h' are related by $\rightarrow_a$ in $s_0 \otimes \alpha^n$;
- $p(h)$ whenever in the unique model $s_0 \otimes \alpha^n$ history h belongs to, we have $p \in \mathsf{V}(h)$;
- and finally, $\mathbf{o}(h)$ whenever h is designated history.

A DEL structure is *propositional* if it is induced by a propositional DEL presentation (s_0, α) where α is propositional.

Figure 12 shows the shape of a DEL structure generated by the online tool *Hintikka's world* [41][3]. Downward dashed arrows represent S_e-transitions (here there are two events named e and f), while solid ones correspond to epistemic relations of two distinct agents. The initial epistemic state s_0 is depicted at the top. It has two worlds with the designated one highlighted in bold. Each world is the root of a tree, thus making DEL structures be forests. The remaining worlds are those of the successive updates $s_0 \otimes \alpha$, $s_0 \otimes \alpha^2$ and $s_0 \otimes \alpha^3$, with some highlighted in bold, as designated states. Every branch in this forest is a history, and those with only bold nodes are designated histories. Some expert readers may view DEL structures as *interpreted system*, or more generally, as *epistemic temporal logic models* [44].

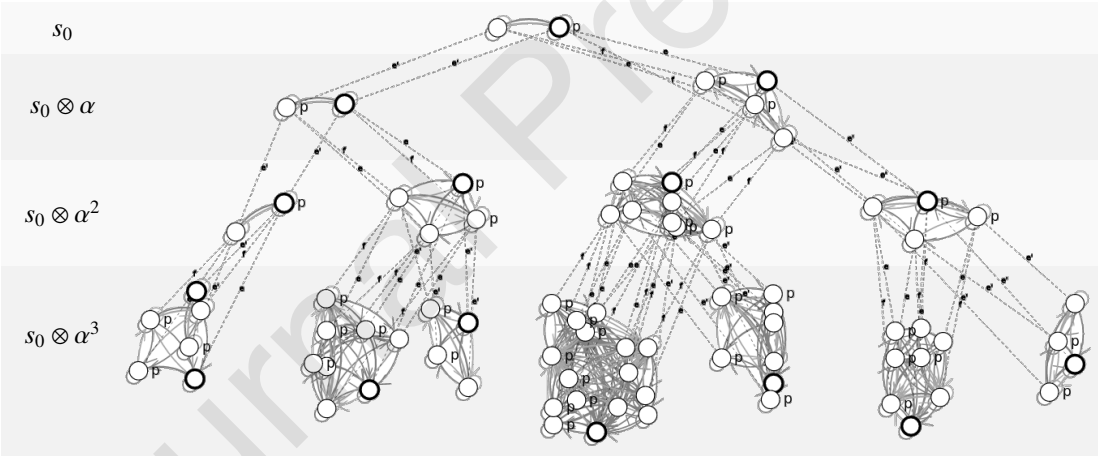
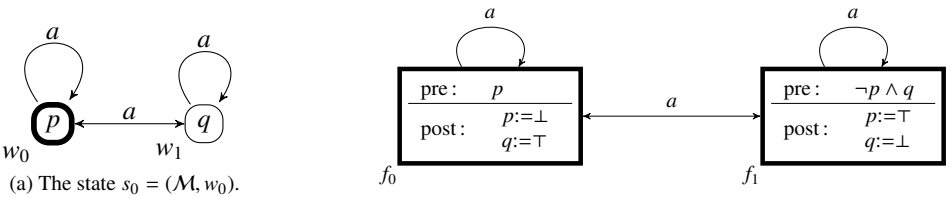


Figure 12: Example of a DEL structure (picture up to level 3).

We now investigate remarkable properties of propositional DEL structures.

Theorem 7. *Propositional DEL structures are automatic structures, and their presentation is effectively computable from* (s_0, α)*.*

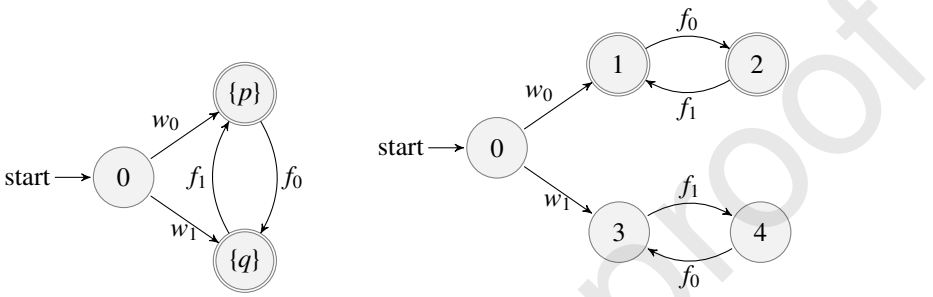
The proof of Theorem 7 relies on the central property that the domain of propositional DEL structures, that is the set of its histories, is a regular language, and that choosing the very identity encoding function allows us to provide finite-state automata for each relation of these structures. Rather than providing the full proof of Theorem 7 that may be found in Maubert [32, Lemma 22, p. 109], we illustrate the construction on a small example.



(a) The state $s_0 = (\mathcal{M}, w_0)$.

(b) The action $(\alpha, \{f_0, f_1\})$.

Figure 13: A simple propositional DEL presentation (s_0, α) with designated events f_0 and f_1.



(a) The automaton $\mathcal{A}_{\mathcal{H}}$.

(b) Automaton that accepts designated histories.

Figure 14: Automata for histories and designated histories for the automatic structure $s_0\alpha_0^*$.

Example 9. Consider the DEL presentation (s_0, α) of Figure 13 that uses atomic propositions of $AP' = \{p, q\}$. We can view the precondition $\mathrm{pre}(e)$ of an event e as the set of all propositional valuations over AP' that satisfy $\mathrm{pre}(e)$. Moreover, we can see a propositional valuation v as the subset of propositions from AP' that are set to true. For instance, $\mathrm{pre}(f_0) = p$ is represented as $\{\{p\}, \{p, q\}\}$. Given a valuation $\mathsf{v} \subseteq AP'$, we write $\mathsf{v} \otimes \mathrm{post}(e)$ for the new valuation obtained after the postcondition of e has been applied. For example, $\{p\} \otimes \mathrm{post}(f_0) = \{q\}$.

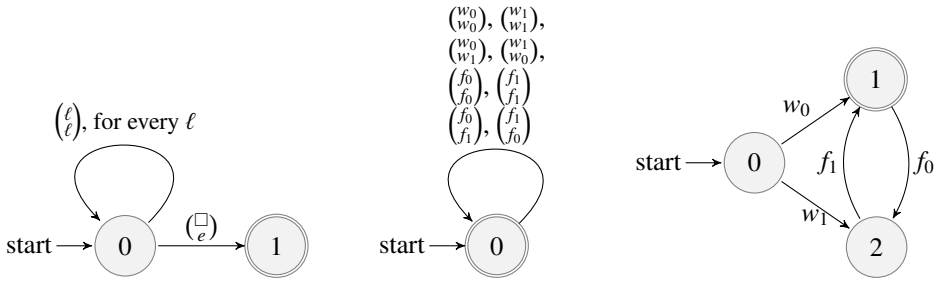
We now turn to the encoding of the elements of the DEL structure. Histories of the DEL structure for the presentation (s_0, α) of Figure 13 are elements of the formal language $\{w_0, w_1\}(\{f_0, f_1\})^*$. In order to encode the relational structure $(\mathcal{H}, S_{f_0}, S_{f_1}, \to_a, p, q, \mathbf{o})$ (see Definition 12), we resort to the finite alphabet $\Sigma = \{w_0, w_1\} \cup \{f_0, f_1\}$ and use the encoding of a history as such. Namely, we consider the identity encoding function $\mathtt{enc}(h) = h$, for every $h \in \mathcal{H}$.

We exhibit a finite-state automaton $\mathcal{A}_{\mathcal{H}}$ whose language is the image of enc, namely the set $\mathcal{H}$ itself. It is drawn in Figure 14a. Automaton $\mathcal{A}_{\mathcal{H}}$ has initial state 0 and other states that are valuations over AP'; thus it is finite. Moreover, all states are accepting but state 0. Regarding the automaton transitions, letter w_0 or w_1 is expected in state 0. When reading a world letter w, automaton moves from state 0 to valuation $\mathsf{V}(w)$. Then, from any valuation v, the automaton expects to read an event letter e with $\mathsf{v} \in \mathrm{pre}(e)$, that makes it move to the updated valuation $\mathsf{v} \otimes \mathrm{post}(e)$. One can easily show that a word is in $L(\mathcal{A}_{\mathcal{H}})$ iff it is a history of the DEL structure. Among histories, some are *designated*. We can build on automaton $\mathcal{A}_{\mathcal{H}}$ to verify whether each read letter is a designated object and obtain the automaton $\mathcal{A}_{\mathbf{o}}$ that accepts all designated histories, thus capturing the unary relation $\mathbf{o}$ of the structure $s_0\alpha^*$, as depicted in Figure 14b.

Regarding the remaining relations of the relational structure $s_0\alpha^*$, we need to build automata for each *e-successor* binary relation S_e, the a-accessibility relation $\to_a$, and unary relations p for $p \in AP'$. Beforehand, we use $\mathcal{A}_{\mathcal{H}}$ in order to build automaton $\mathcal{A}_{\mathcal{H}\times\mathcal{H}}$ that accepts the "full" binary relation $\mathcal{H} \times \mathcal{H}$. Now, relation S_e is captured by the synchronous product of automaton $\mathcal{A}_{\mathcal{H}\times\mathcal{H}}$ and of an automaton that accepts all pairs of histories of the form (h, he), depicted in Figure 15a. Similary, relation $\to_a$ is represented by the product automaton between $\mathcal{A}_{\mathcal{H}\times\mathcal{H}}$ and automaton $\mathcal{A}_{\to_a}$ of Figure 15b. Finally, the unary relation p is represented by the automaton $\mathcal{A}_p$ of Figure 15c.

As can be seen in Lemma 22, p. 109, of Maubert [32] and in the example above, in the proof of Theorem 7, the encoding of the elements (the histories) of the DEL structure is the identity function. As a consequence, the encoding

[3] http://hintikkasworld.irisa.fr/



(a) Automaton $\mathcal{A}_{S_e}$. (b) Automaton $\mathcal{A}_{\rightarrow_a}$. (c) Automaton $\mathcal{A}_p$.

Figure 15: Some automata for the automatic structure $s_0\alpha_0^*$.

of a history displays the initial word where this history starts followed by the sequence of all performed events in order along this history. This identity encoding makes the automaton of the goal formula a recogniser of the set of all solutions of the planning task, that is, all action sequences that achieve the goal from the initial state. We explain how in the next paragraph.

The plan existence problem and the plan synthesis. When considering the goal formula φ_g of a planning task, one effectively builds a *plan automaton* that recognizes exactly all plans achieving φ_g. The algorithm that computes this automaton is as follows. Given a planning task $T = (s_0, \alpha, \varphi_g)$:

1. Compute the FO-formula $\Psi_g(x)$ equivalent to φ_g (By the translation of epistemic logic into FO[4], formula Ψ_g indeed has a single free first-order variable);
2. Compute $\mathcal{A}_{\Psi_g}$ (Proposition 6) which accepts all histories in (s_0, α) (Definition 12) that satisfy Ψ_g, or equivalently φ_g.
3. Intersect automaton $\mathcal{A}_{\Psi_g}$ with automaton $\mathcal{A}_{\mathbf{o}}$ (Figure 14b) in order to restrict its language to designated histories $we_1 \dots e_n$. Write the resulting automaton $\mathcal{A}_{(s_0,\alpha,\varphi_g)}$.

We can now finalize the proof of Theorem 4: the effectively constructed automaton $\mathcal{A}_{(s_0,\alpha,\varphi_g)}$ can be used to solve the epistemic plan existence problem: if the language of $\mathcal{A}_{(s_0,\alpha,\varphi_g)}$ is empty, then the answer is "No" for the instance (s_0, α, φ_g), otherwise the answer is "Yes", and by a reachability analysis in this automaton, one can synthesize a solution to the planning task.

Additionally, automaton $\mathcal{A}_{(s_0,\alpha,\varphi_g)}$ contains a lot of information that can be further exploited. For example, one can decide if there are infinitely many plans, since one can decide if a finite-state automaton accepts an infinite language. Also, one may use automaton $\mathcal{A}_{(s_0,\alpha,\varphi_g)}$ to search a shortest solution, or more generally an optimal solution, provided the automaton is made "weighted" by giving a cost function on events.

4.1.2. The case of propositional preconditions and no postconditions

Theorem 8 ([15])**.** *PlanEx-$\mathcal{T}(0,-1)$ is PSPACE-complete.*

Proof. Originally, PlanEx-$\mathcal{T}(0,-1)$ was proven to be EXPSPACE [12], but the result was then strengthened to PSPACE-completeness [15]. Both proofs use a fundamental property of $\mathcal{T}(0,-1)$, namely that actions with propositional preconditions and without postconditions commute: for all epistemic states s and epistemic actions α and α' in $\mathcal{T}(0,-1)$, $s \otimes \alpha \otimes \alpha'$ and $s \otimes \alpha' \otimes \alpha$ satisfy the same epistemic formulas [30]. Furthermore, when a modal depth d is fixed (typically, the modal depth of the goal φ_g), we know that for all epistemic states s, $s \otimes \alpha^{|\alpha|^d}$ and $s \otimes \alpha^{|\alpha|^d+1}$ satisfy

[4] see Appendix C.

the same epistemic formulas of modal depth at most d [39]. In particular, it means that when searching for a solution to a planning task in $\mathcal{T}(0,-1)$, we never have to apply an action α more than $|\alpha|^d$ times where $d = md(\varphi_g)$.

Given these two properties, the algorithm is defined as follows for any planning task $T = (s_0, A, \varphi_g)$ with $A = \{\alpha_1, \ldots, \alpha_m\}$.

1. First, compute $d = md(\varphi_g)$.
2. Second, non-deterministically guess $n_1, \ldots, n_m$ such that $n_i \in 0, \ldots, |\alpha_i|^d$.
3. Third, check that $\alpha_1^{n_1}, \ldots, \alpha_m^{n_m}$ is a solution to T.

Since $n_1, \ldots, n_m$ are written in binary, the guesses are performed in polynomial space. The third point is the hardest part of the proof, since it must be done in polynomial space. The idea is to develop a model checking procedure where the worlds of the model $s_0 \otimes \alpha_1^{k_1} \otimes \cdots \otimes \alpha_m^{k_m}$ with $k_i \leq n_i$ are represented by tuples $(k_1, \ldots, k_m)$, which are stored in polynomial space. The algorithm is thus running non-deterministically in polynomial space. Since NPSPACE = PSPACE [40], the result is proved.

The PSPACE lower bound is by reduction from True Quantified Binary Formula (TQBF). The proof can be found in Bolander et al., 2015 [12, Th. 5.3]. □

4.2. Undecidable cases

In Section 4.2.1 we will show that PlanEx-$\mathcal{T}(1,0)$ is undecidable, and can hence conclude that all PlanEx-$\mathcal{T}(m,n)$ with either $m \geq 1, n \geq 0$ or $m \geq 0, n \geq 1$ are undecidable, using Theorems 1 and 2 (see again Figure 9 for an illustration of this). Since PlanEx-$\mathcal{T}(0,0)$ has already been proved decidable, the only remaining cases are PlanEx-$\mathcal{T}(n,-1)$ for $n \geq 0$. In Section 4.2.2 we detail the undecidability of one of these remaining cases, PlanEx-$\mathcal{T}(2,-1)$.

4.2.1. The case of precondition of modal depth 1

In 2011, Bolander and Andersen [11] proved that the plan existence problem PlanEx-$\mathcal{T}(1,0)$ is undecidable. Their proof was done by a reduction from the halting problem of a Turing machine to PlanEx-$\mathcal{T}(1,0)$. More recently, Lê Cong et al. [16] proposed a proof that PlanEx-$\mathcal{T}(1,1)$ is undecidable, and because PlanEx-$\mathcal{T}(1,1) \leq^P$ PlanEx-$\mathcal{T}(1,0)$ (see Theorem 3), so is PlanEx-$\mathcal{T}(1,0)$. The proof of Lê Cong et al. relies on a cellular automaton instead of a Turing machine, but remains close to the one of Bolander and Andersen: a configuration of the machine (Turing machine or cellular automaton) is encoded by an epistemic state; the initial configuration is encoded by the initial epistemic state; epistemic actions simulate steps of computation; the goal formula specifies that the configuration is halting. The advantages of the proof of Lê Cong et al. are the following. First, cellular automata are less cumbersome than Turing machines: no need to handle left-going and right-going transitions. Second, their proof gives small conditions on the planning tasks for which the plan existence problem is already undecidable: two agents, **S5** models, and only 6 atomic propositions are sufficient to get undecidability. More importantly, it is undecidable even if the actions and the goal are also fixed. Intuitively, it means that there is a fixed domain for which the plan existence problem is already undecidable (just the initial epistemic state may vary). For these reasons, we sketch the proof of Lê Cong et al.

Theorem 9 ([16]). *PlanEx-$\mathcal{T}(1,1)$ is undecidable, even if we restrict to **S5** models, 2 agents, 3 actions, and 6 atomic propositions.*

We sketch the proof in the following. Before giving the reduction, we recall some background on (one-dimensional three-cell neighborhood) cellular automata. An infinite sequence of cells are settled on a line (tape); each cell is in a state represented by a *symbol*[5] of a finite alphabet Σ. A transition function f maps a three-cell neighborhood (left-cell symbol, current symbol, right-cell symbol) to the new symbol of the cell. Formally, a *cellular automaton* is a pair $\mathcal{A} = (\Sigma, f)$ where Σ is a finite alphabet and $f : \Sigma^3 \to \Sigma$ is a (partial) transition function. For instance, the Rule 110 cellular automaton [48] is the two-symbol cellular automaton $\mathcal{A}_{\text{R110}} = (\{0,1\}, f_{110})$ where f_{110} is defined by the propositional formula $f_{110}(x,y,z) := (x \wedge y \wedge \neg z) \vee (x \wedge \neg y \wedge z) \vee (\neg x \wedge y \wedge z) \vee (\neg x \wedge y \wedge \neg z) \vee (\neg x \wedge \neg y \wedge z)$.

[5] We use 'symbol' instead of 'cell state', to avoid confusion with a knowledge state.

Figure 16: Rule 110 transition function ($f_{110}(1,0,1) = 1$, etc.) and some successive configurations.

A *configuration*, that is, the symbols of cells on an infinite line, is modeled by an infinite word $c \in \Sigma^{\mathbb{Z}}$, that is, a map that assigns a symbol $c[i]$ to any integer $i \in \mathbb{Z}$. A computation step is performed by the following rule. Given a cellular automaton $\mathcal{A}$ and an infinite word $c \in \Sigma^{\mathbb{Z}}$, we define the successor of c by $\mathcal{A}$ to be the infinite word c' defined by $c'[i] := f(c[i-1], c[i], c[i+1])$. We write $c \to_{\mathcal{A}} c'$. It may happen that the successor is not defined since f may be partial. Figure 16 shows the transition function f_{110} graphically and some successive configurations.

A cellular automaton is deemed *universal* if it can simulate any Turing machine; the quest for finding such *small* universal cellular automata started in the 1960s. A common hypothesis is to assume a *blank background*: we consider that alphabets always contain a special symbol $\sqcup$ and that transition functions map $\sqcup\sqcup\sqcup$ to $\sqcup$. Furthermore, we assume that configurations are *finite*, in the sense that almost all cell symbols are $\sqcup$ except a finite number; configurations are of the form $\sqcup^\omega u \sqcup^\omega$ where u is a finite word, called the *support*. Starting from a finite configuration only leads to finite configurations, after a finite number of computation steps.

Smith proved in 1968 [42, Theorem 40] that any m-symbol n-state Turing machine can be simulated by a $(m+2n)$-symbol[6] cellular automaton with a blank background. As Minsky constructed a 4-symbol 7-state universal Turing machine M_{Minsky} [33], there exists a $4+2\times 7 = 18$-symbol universal cellular automaton $\mathcal{A}_{\text{Smith}} = (\Sigma_{\text{Smith}}, f_{\text{Smith}})$, that simulates M_{Minsky}. As a consequence, there exists a finite word[7] h_{Smith}, such that following decision problem called CellularAutomataReach, is undecidable:

> Given a finite word u, decide whether $\sqcup^\omega u \sqcup^\omega \to^*_{\mathcal{A}_{\text{Smith}}} c$ where the configuration c contains the pattern h_{Smith}.

The rest of our proof sketch describes a reduction from CellularAutomataReach into PlanEx-$\mathcal{T}(1,0)$. This is achieved by simulating executions of cellular automata with blank backgrounds.

Encoding cellular automaton configurations in epistemic states. First of all, we introduce sufficiently many propositions to encode symbols of the alphabet. For alphabet Σ_{Smith} with 18 symbols that we respectively write $\ell_0, \ell_1, \ldots, \ell_{17}$, only 5 propositions $p_1, \ldots, p_5$ suffice. Given a symbol ℓ, we note $\mathsf{enc}(\ell)$ the encoding of ℓ: $\mathsf{enc}(\ell_0) = \neg p_1 \wedge \ldots \neg p_5$, $\mathsf{enc}(\ell_1) = p_1 \wedge \neg p_2 \ldots \neg p_5$, etc. Without loss of generality, we suppose that symbol $\sqcup$ is ℓ_0 and is encoded by the valuation making all p_i false. In the remainder, $\vec{p}$ denotes the sequence of propositions p_i's.

Now we encode the finite supports of configurations by means of so-called *finite linear states* [16]—such states appear in real epistemic puzzles such as the consecutive number puzzle presented in Example 6. For odd $n > 0$, we define a *finite linear state bounded by n* as an epistemic state of the form $((\{-n, \ldots, -1, 0, 1, \ldots n\}, (\to_a)_{a\in Ag}, \mathsf{V}), 0)$ where:

1. $\to_a = \{(k,k) \mid k \in [\![-n; n]\!]\} \cup \left\{(2k, 2k+1), (2k+1, 2k) \mid \frac{-n+1}{2} \leq k \leq \frac{n-1}{2}\right\}$;
2. $\to_b = \{(k,k) \mid k \in [\![-n; n]\!]\} \cup \left\{(2k, 2k-1), (2k-1, 2k) \mid \frac{-n+1}{2} \leq k \leq \frac{n-1}{2}\right\}$;
3. $\heartsuit \in \mathsf{V}(k)$ iff k is even;
4. for all i, $p_i \notin \mathsf{V}(-n), \mathsf{V}(-n+1), \mathsf{V}(n-1), \mathsf{V}(n)$.

[6] Referred to as $(m+2n)$-state by Smith [42].

[7] According to the notation of Table 14.8-1 p. 279 in Minsky, 1967 [33], word h_{Smith} is $q_3 0$, with our notation, it is $3\sqcup$.

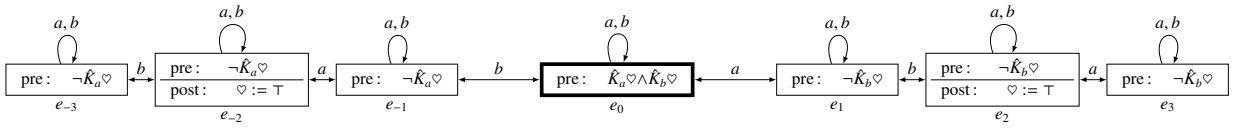
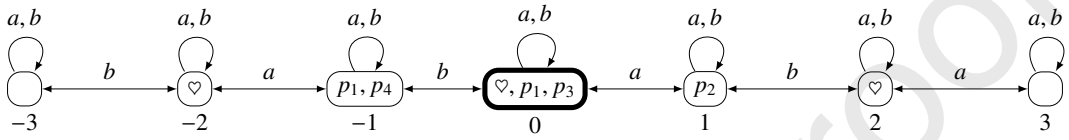


Figure 17: Skeleton of action F.

A finite linear state bounded by n then encodes a finite configuration c whose support is of length smaller than $2n-3$ when $V(k)$ makes $\mathsf{enc}(\ell_i)$ true iff $c[k]=\ell_i$; we require that the two terminal ($-n$ and n) and the two pre-terminal ($n-1$, $-n+1$) worlds encode $\textvisiblespace$ (Condition 4). For instance, the $\mathcal{A}_{\text{Smith}}$ configuration $\textvisiblespace^{\omega}\textvisiblespace\textvisiblespace\ell_9\ell_5\ell_2\textvisiblespace\textvisiblespace\textvisiblespace^{\omega}$ where $c[-1]=9$ can be represented by:



Given a finite word u, let us set once and for all the finite linear state s_u used to encode u: substituting $u\textvisiblespace^m$ for u with $m\in\{0,1,2,3\}$ so that $|u|$ will be congruent to 3 modulo 4, we let s_u be the finite linear state bounded by $\frac{|u|+3}{2}$ where $\mathsf{V}(k-\frac{|u|-1}{2})$ makes $\mathsf{enc}(\ell_i)$ true iff $u[k]=\ell_i$ for $0\le k\le|u|-1$, and makes $\mathsf{enc}(\textvisiblespace)$ true outside of this range. These are only technicalities ensuring a sufficiently large odd index for the interval state to respect constraints with a pseudo-centered word.

Simulating cellular automata in DEL. We define an action F mimicking one computation step of the cellular automaton: if s is a finite linear state encoding a configuration c, then $s\otimes F$ is (isomorphic to) a finite linear state encoding the successor of c. Action F is partially given by Figure 17. Intuitively, the actual event e_0 copies every non-terminal world; event e_{-1} keeps the left-tip world, while e_{-2} and e_{-3} clone it to append two new worlds to the left. Events e_1, e_2, e_3 play a similar part. In the end, action F adds two new worlds on each side, while preserving the canonical knowledge state structure that we aim for, including the tips' asymmetry relatively to the agents.

We finish the definition of F by adding postconditions for the p_j, corresponding to the application of a transition function f. Suppose without loss of generality that $\heartsuit$ holds in a given world $k\in\{-n+1,\dots,n-1\}$. Bits of $c[k-1]$ are obtained by taking the b-transition to the $\neg\heartsuit$-world of world k. The i^{th} bit of $c[k-1]$ is given by $\hat{K}_b(\neg\heartsuit\wedge p_i)$. In the following we use the notation $\langle\hat{K}_b(\neg\heartsuit\wedge p_i)\rangle_i$ to denote the sequence of formulas $\hat{K}_b(\neg\heartsuit\wedge p_i)$. In the same way, the i^{th} bit of $c[k+1]$ is given by $\hat{K}_a(\neg\heartsuit\wedge p_i)$. The case where $\heartsuit$ does not hold in the current world is symmetric. We model f by propositional formulas $f_j(\vec{p}^{\,-},\vec{p},\vec{p}^{\,+})$ over three sequences of atomic propositions $\vec{p}^{\,-}$ (left cell symbol), $\vec{p}$ (middle cell symbol), $\vec{p}^{\,+}$ (right cell symbol) that return the value of the j^{th} bit of the new symbol at the middle cell. The postconditions for the p_j in F are thus defined as follows. First, $\text{post}(e_k)(p_j)=\bot$ for all $k\neq 0$. Event e_0 effectively applies f where $\text{post}(e_0)(p_j)$ is the formula

$$\Big(\heartsuit\to f_j\Big(\langle\hat{K}_b(\neg\heartsuit\wedge p_i)\rangle_i,\vec{p},\langle\hat{K}_a(\neg\heartsuit\wedge p_i)\rangle_i\Big)\Big)\wedge\Big(\neg\heartsuit\to f_j\Big(\langle\hat{K}_a(\heartsuit\wedge p_i\rangle_i),\vec{p},\langle\hat{K}_b(\heartsuit\wedge p_i)\rangle_i\Big)\Big).$$

We will refer to F_{Smith} for the epistemic action that corresponds to the transition function f_{Smith} of the cellular automaton $\mathcal{A}_{\text{Smith}}$.

On top of proposition $\heartsuit$, when considering the particular cellular automaton $\mathcal{A}_{\text{Smith}}$, we need no more than 5 extra propositions to encode all symbols of alphabet Σ_{Smith} of cardinality 18, so that an overall set of 6 propositions suffices.

The interested reader can run simulations in DEL of cellular automata using the online software *Hintikka's world*. Figure 18 shows a screenshot of Hintikka's World depicting a finite linear state.

Detecting the pattern h_{Smith}. The pattern h_{Smith} can be located far from the designated world in the current state. That is why we introduce two more actions R_{Smith} and L_{Smith} that shifts the cells, while still enforcing growth of the state in order to avoid information overflow on each side. The two new actions are defined below:

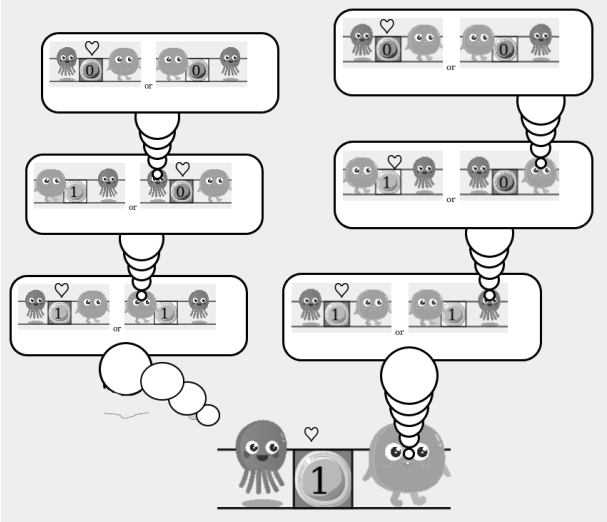


Figure 18: The finite linear state encoding the configuration 0011100. The real world corresponds to the central cell 1. Agent b (on the right in the real world) imagines a world that corresponds to the right cell 1 in which agent a imagines a world that corresponds to the right-right cell 0, etc.

- R_{Smith} (*right shift*) is defined by the same structure as F_{Smith} is, except for the following postconditions on event e_0: for all i, the assignment of p_i is replaced by $p_i := (\heartsuit \wedge \hat{K}_b(\neg\heartsuit \wedge p_i)) \vee (\neg\heartsuit \wedge \hat{K}_a(\heartsuit \wedge p_i))$;
- L_{Smith} (*left shift*) is akin to R_{Smith}; let us just give the postcondition for p_i on e_0 since nothing else differs: $p_i := (\heartsuit \wedge \hat{K}_a(\neg\heartsuit \wedge p_i)) \vee (\neg\heartsuit \wedge \hat{K}_b(\heartsuit \wedge p_i))$.

Now we define formulas encoding the occurrence of a pattern h in the configuration. We first define the formula $\mathsf{wenc}(h) := \mathsf{wenc}_{\heartsuit}(h) \vee \mathsf{wenc}_{\neg\heartsuit}(h)$ where formulas $\mathsf{wenc}_{\heartsuit}(h)$ and $\mathsf{wenc}_{\neg\heartsuit}(h)$ are inductively defined by:

- $\mathsf{wenc}_{\heartsuit}(\varepsilon) = \heartsuit$ and $\mathsf{wenc}_{\neg\heartsuit}(\varepsilon) = \neg\heartsuit$;
- for all letters ℓ, $\mathsf{wenc}_{\heartsuit}(\ell h) = \heartsuit \wedge \mathsf{enc}(\ell) \wedge \hat{K}_a \mathsf{wenc}_{\neg\heartsuit}(h)$;
- for all letters ℓ, $\mathsf{wenc}_{\neg\heartsuit}(\ell h) = \neg\heartsuit \wedge \mathsf{enc}(\ell) \wedge \hat{K}_b \mathsf{wenc}_{\heartsuit}(h)$.

From a CellularAutomataReach-instance u, we compute the equivalent planning task: the initial epistemic state is s_u; the set of actions is $\{F_{\text{Smith}}, L_{\text{Smith}}, R_{\text{Smith}}\}$; the goal formula is $\mathsf{wenc}(h_{\text{Smith}})$. Note that the set of actions and the goal formula do not depend on u. This completes the sketch of the construction for proving Theorem 9. The following then follows directly from Theorems 3, 1 and 9.

Corollary 10. *PlanEx-$\mathcal{T}(1,0)$ is undecidable, as well as each problem PlanEx-$\mathcal{T}(m,n)$ with $m \geq 1$ and $n \geq 0$.*

We can benefit from the undecidability result for PlanEx-$\mathcal{T}(1,1)$ given in Theorem 9 and from the reduction provided by Theorem 3 for the particular case PlanEx-$\mathcal{T}(1,1) \leq^P$ PlanEx-$\mathcal{T}(1,0)$ to draw conclusions on PlanEx-$\mathcal{T}(1,0)$: it remains undecidable for S5 models, 2 agents, and for some fixed bounded values on the number of agents, of actions, and of propositions. We however believe that the obtained bounds via the two reductions CellularAutomataReach $\leq^P$ PlanEx-$\mathcal{T}(1,1) \leq^P$ PlanEx-$\mathcal{T}(1,0)$ can be improved.

4.2.2. The case of preconditions of modal depth 2 and no postconditions

Aucher and Bolander [2] proved that PlanEx-$\mathcal{T}(\infty,-1)$ is undecidable, using a reduction from the halting problem for two-counter machines known to be undecidable [33] (the ∞ in the first argument of $\mathcal{T}$ means that the proof didn't put any constraints on the modal depth of preconditions). Charrier et al. [15] strengthened the construction of that proof in order to achieve the following.

Theorem 11 ([15]). *PlanEx-$\mathcal{T}(2,-1)$ is undecidable.*

Proof. We begin by defining two-counter machines and then explain roughly the idea of the proof by Charrier et al. [15]. Formally, a two-counter machine M is a sequence of instructions $(I_0, \ldots, I_N)$ where:

- For $\ell < N$, I_ℓ is either $inc(i)$, $goto(\ell')$ or $gotocond(i, \ell')$, with $i \in \{1, 2\}$, $\ell' \leq N$ and $\ell \neq \ell'$;
- $I_N = halt$.

A configuration of a two-counter machine M is a triple (ℓ, c_1, c_2) where $\ell \in \{0, \ldots, N\}$ is the *program counter* and $c_1, c_2 \in \mathbb{N}$ are the two *data counters*. Instruction $inc(i)$ increments data counter i, instruction $goto(\ell)$ jumps to instruction I_ℓ and instruction $gotocond(i, \ell)$ jumps to instruction I_ℓ only if data counter i has value 0, and decrements data counter i otherwise.

The set $C_M = \{0, \ldots, N\} \times \mathbb{N} \times \mathbb{N}$ is thus the set of all potential configurations.

The transition function $\rightarrow_M$ on C_M is defined as follows. For all $(\ell, c_1, c_2) \in C_M$:

- If $I_\ell = inc(1)$, $(\ell, c_1, c_2) \rightarrow_M (\ell + 1, c_1 + 1, c_2)$;
- If $I_\ell = inc(2)$, $(\ell, c_1, c_2) \rightarrow_M (\ell + 1, c_1, c_2 + 1)$;
- If $I_\ell = goto(\ell')$, $(\ell, c_1, c_2) \rightarrow_M (\ell', c_1, c_2)$;
- If $I_\ell = gotocond(1, \ell')$, $(\ell, c_1, c_2) \rightarrow_M \begin{cases} (\ell', 0, c_2) & \text{if } c_1 = 0; \\ (\ell + 1, c_1 - 1, c_2) & \text{otherwise;} \end{cases}$
- If $I_\ell = gotocond(2, \ell')$, $(\ell, c_1, c_2) \rightarrow_M \begin{cases} (\ell', c_1, 0) & \text{if } c_2 = 0; \\ (\ell + 1, c_1, c_2 - 1) & \text{otherwise.} \end{cases}$

A two-counter machine M *halts* if there exist c_1, c_2 such that $(0, 0, 0) \rightarrow_M^* (N, c_1, c_2)$, where $\rightarrow_M^*$ denotes the reflexive transitive closure of $\rightarrow_M$. The halting problem for two-counter machines consists in deciding, given a two-counter machine, whether it halts or not. This problem is known to be undecidable [33].

The reduction. Charrier et al. [15] define an effective reduction r that, given a two-counter machine M, computes an epistemic planning task $r(M)$ of PlanEx-$\mathcal{T}(2,-1)$. Only one agent is needed, so K_a is noted K and $\hat{K}_a$ is noted $\hat{K}$. To ease the reading of the figures, we do not name the worlds.

The initial epistemic state represents the initial configuration $(0, 0, 0)$ drawn in Figure 19, whereas Figure 20 depicts the epistemic state representing the configuration $(1, 0, 2)$. It is composed of three zones:

- One for the program counter (PC) such that the program counter is $\ell = i$ if the only world in this zone having a successor is the one labeled by a_i. Initially, $\ell = 0$ so the only world having a successor is the one labeled by a_0.
- One for each data counter, c_1 and c_2. Each zone contains two worlds labeled by p_i and q_i with $i \in \{1, 2\}$. The world labeled by q_i marks the beginning of the chain representing the integer stored in c_i, and the chain is made of worlds labeled by p_i. Initially, $c_i = 0$, so the chain is made of one world labeled by p_i.

Each program line $\ell{:}I_\ell$ is represented by some action. We do not detail every instruction here but give examples. The action for the instruction $goto(\ell')$ at program counter ℓ is given in Figure 22 and uses $repl(\ell, \ell')$ given in Figure 21. In the program counter zone, it removes the successor for the world labeled by a_ℓ (with the precondition $a_\ell \wedge \hat{K}K\bot$) and creates a successor for the world labeled by $a_{\ell'}$ (with the events having precondition $a_{\ell'}$). In the data counter zones, the action does not change anything.

Another example is $inc(1)$ given in Figure 24, using the portion $repl(\ell, \ell')$ and the portion $lengthen(1)$ of Figure 23. It replaces the program counter ℓ with $\ell + 1$ and adds a world labeled by p_1 in the c_1 portion of the Kripke model.

The goal formula checks that the epistemic state is in program counter N, that is, $\varphi_g = \bigwedge_{\ell=0}^{N} \hat{K}a_\ell$.

□

Combining Corollary 10 and Theorem 11, we can conclude that the only decision problem PlanEx-$\mathcal{T}(m, n)$ for which decidability or undecidability has not yet been established is PlanEx-$\mathcal{T}(1,-1)$. Unfortunately, this case remains open. We discuss some advances towards solving this open problem in the next section.

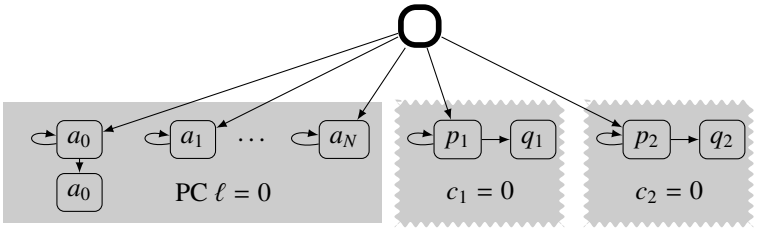


Figure 19: Epistemic state representing the initial configuration $(0, 0, 0)$.

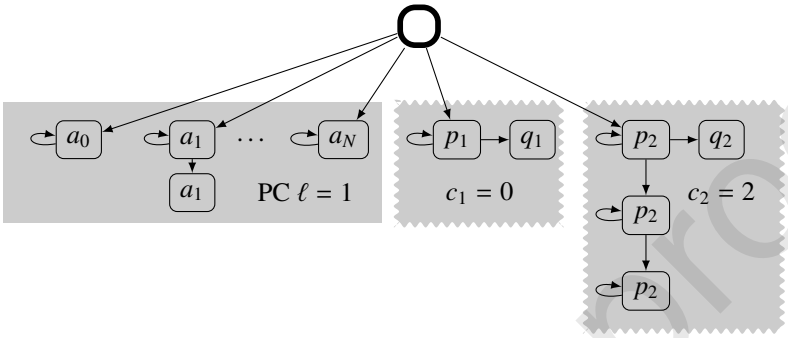


Figure 20: Epistemic state representing the configuration $(1, 0, 2)$.

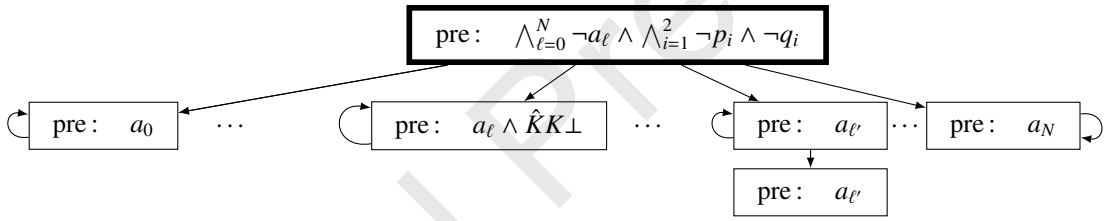


Figure 21: Action portion $repl(\ell, \ell')$ for $\ell \neq \ell'$.

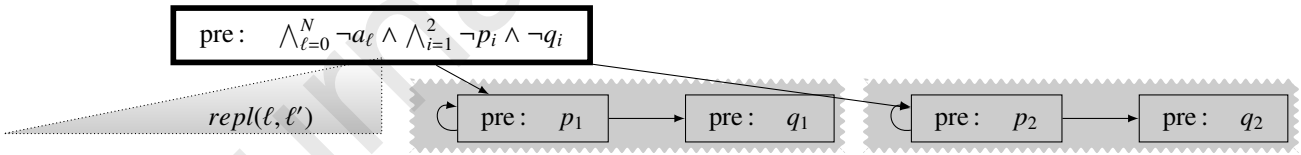


Figure 22: Action $\alpha_{\ell:goto(\ell')}$ for ℓ:$goto(\ell')$.

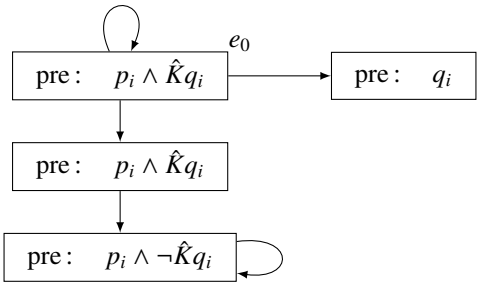


Figure 23: Action portions *lengthen*(i).

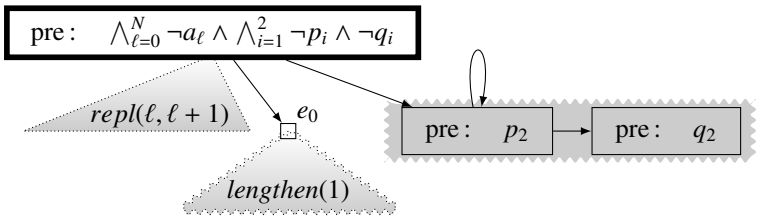


Figure 24: Action $\alpha_{\ell:inc(1)}$ for ℓ:$inc(1)$.

4.3. The open case of actions with preconditions of modal depth 1 *and no postconditions*

The decidability of PlanEx-$\mathcal{T}(1,-1)$ is still an open problem. There are some hints that the DEL structure (see Definition 12) may not be automatic as for PlanEx-$\mathcal{T}(0,0)$ [20]. In other words, the technique used in Theorem 4 to prove that PlanEx-$\mathcal{T}(0,0)$ is decidable may be not suitable to prove that PlanEx-$\mathcal{T}(1,-1)$ is.

The problem PlanEx-$\mathcal{T}(1,-1)$ may also be undecidable. At least when extending the actions from the decidable class $\mathcal{T}(0,0)$ with the actions from the class $\mathcal{T}(1,-1)$ we already get an undecidable class (a class with an undecidable plan existence problem). In other words, PlanEx-$\mathcal{T}((0,0)\cup(1,-1))$ is undecidable, where $\mathcal{T}((0,0)\cup(1,-1))$ is the class of planning tasks in which each action either belongs to $\mathcal{T}(0,0)$ or $\mathcal{T}(1,-1)$. Even worse, we will prove that just adding public announcements of modal depth 1 to $\mathcal{T}(0,0)$ is sufficient to make the corresponding plan existence problem undecidable. To prove this undecidability result, we provide in Appendix D a reduction from PlanEx-$\mathcal{T}(1,0)$ to PlanEx-$\mathcal{T}((0,0)\cup(1,-1))$. The reduction is quite technical although the overall idea is simple. Each action α in the original plan is simulated by the execution of three actions:

1. a propositional action that creates a copy of each world augmented with a possible valuation ν of fresh propositional variables p_ψ, one for each precondition appearing in α;
2. the public announcement that, for every preconditions ψ of α, formula $p_\psi \leftrightarrow \psi$ holds, said differently the public announcement that rules out all augmented worlds where valuation ν has not properly set some proposition p_ψ;
3. a copy of α that is a propositional action, but where each precondition ψ of α is replaced by p_ψ.

Theorem 12. *PlanEx-$\mathcal{T}(1,0) \leq^P$ PlanEx-$\mathcal{T}((0,0)\cup(1,-1))$. Actually, PlanEx-$\mathcal{T}(1,0)$ reduces to the restriction of PlanEx-$\mathcal{T}((0,0)\cup(1,-1))$ in which each action either has propositional pre- and post-conditions, or is a public announcement (*i.e. *containing a unique event) of modal depth at most 1.*

Proof. The detailed proof is in Appendix D. □

Corollary 13. *PlanEx-$\mathcal{T}((0,0)\cup(1,-1))$ is undecidable.*

5. PSPACE variants of the plan existence problem

PSPACE can be seen as the new algorithmic challenge, after the class NP. For having efficient algorithms for decision problems in PSPACE, a lot of efforts are put in the design of efficient QBF (quantified binary formulas) solvers (see for instance Marques-Silva [31]).

Theorem 8 showed that epistemic planning for actions with propositional preconditions and no postcondition is PSPACE-complete. In this section, we discuss two important ways to obtain decidability (and below PSPACE!) and to limit the search-space to a finite one. First we can bound the length of solutions (Subsection 5.1): this is called *bounded* epistemic planning. Second, we can focus on epistemic actions whose application does not make epistemic models grow (Subsection 5.2).

5.1. Bounded epistemic planning

In this section, we address the bounded version of the plan existence problem, defined as follows.

Definition 13. By BoundedPlanEx, we denote the following decision problem, called the *bounded plan existence problem*:

> Given a planning task T, given a integer β written in unary, does T have a solution of length smaller than β?

In the previous problem, the bound is written unary as in the so-called polynomial-length plan existence problem [43]. This assumption is reasonable since the planner that solves BoundedPlanEx should at least have allocated enough memory cells to store the output.

Theorem 14. *BoundedPlanEx is PSPACE-complete.*

Proof. PSPACE-membership is proved by reducing in polynomial time BoundedPlanEx to model-checking against the dynamic language of DEL, which is in PSPACE [4]. The dynamic language has a dynamic construction $\langle A\rangle\varphi$, where A is a finite set of actions, as in dynamic logics that says "there is an action α in A that is executable, and after executing it, φ holds'". Let us consider a planning task $T = (s_0, A, \varphi_g)$. Without loss of generality, we suppose that A contains a public skip action (that is, an action made up of one event with a self-loop, whose precondition is $\top$ and with no postcondition). Thus, T has a solution of length smaller than β iff T has a solution of length β iff $s_0 \models \langle A\rangle^\beta\varphi_g$, where $\langle A\rangle^\beta$ is $\langle A\rangle \dots \langle A\rangle$ where the modality $\langle A\rangle$ is repeated β times.

PSPACE-hardness is proved by reducing in polynomial time from the plan existence problem where the initial epistemic state is a chain, actions are tree-like, all preconditions are propositional, and there are no postconditions. The plan existence problem for such planning tasks is PSPACE-hard [12, Th. 5.4]. Actually, if a planning task $T = (s_0, A, \varphi_g)$ being an instance of this plan existence problem has a solution, it has a solution of length at most $\max_{\alpha\in A}(\ell(\alpha))$, where $\ell(\alpha)$ is the number of leaves of the tree α. The reduction maps T to $(T, \max_{\alpha\in A}(\ell(\alpha)))$.

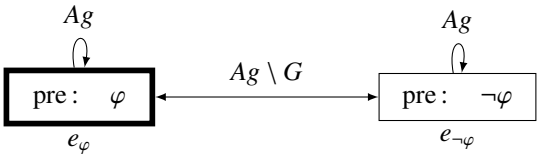
□

5.2. Separable event models

We introduce separable actions, also defined as globally deterministic actions by Bolander and Andersen [11]. The intuition between separable actions is simple: preconditions must be *mutually inconsistent*, defined formally as follows.

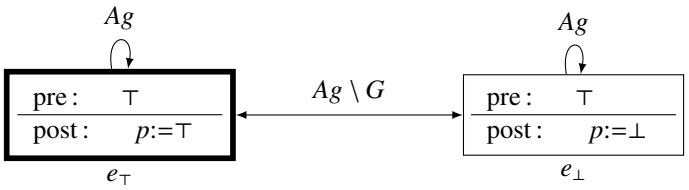
Definition 14. An action $\alpha = (\mathcal{E}, e_0)$ with $\mathcal{E} = (\mathsf{E}, (R_a^{\mathcal{E}})_{a\in Ag}, \mathrm{pre}, \mathrm{post})$ is *separable* if and only if for all events $e, f \in \mathsf{E}$ such that $e \neq f$, the formula $\mathrm{pre}(e) \wedge \mathrm{pre}(f)$ is unsatisfiable.

Example 10. We provide some example of separable and non-separable actions. Actions with a single event are trivially separable. Such actions are usually called *public actions*. Public actions without postconditions are called *public announcements*, as already previously mentioned. General public actions represent a possibly ontic action being executed publicly, so that it is common knowledge among all agents that it has been executed (the postcondition will become common knowledge). *Semi-private announcements* to a group $G \subseteq Ag$ are also separable, since they are defined to be of the following following form:



Such a semi-private announcement represents φ being announced to the agents in G (these agents only consider the event e_φ possible), whereas the agents not in G will not know whether φ or $\neg\varphi$ has been announced (they cannot distinguish events e_φ and $e_{\neg\varphi}$). Public actions and semi-private announcements are for instance considered in the work of Kominis and Geffner [27].

Non-separable actions include *semi-private assignments*, defined by:



Such a semi-private assignment represents p being set true, but the agents not in G doesn't know whether p is set true or false. It is not separable since the preconditions of both events are simply $\top$, and these are clearly not mutually inconsistent.

When actions are separable, the following two facts hold:

- The plan existence problem is NP-complete if all actions are without postconditions.
- The plan existence problem is PSPACE-complete otherwise.

It is proven by Bolander et al. [12] that for actions with one event and without postconditions, the plan existence problem is NP-complete. For actions with one event and with postconditions, it is proven by Jensen [26] that the plan existence problem is PSPACE-complete. Let us concentrate on upper bound results.

Theorem 15. *We consider the plan existence problem under the constraint that A only contains separable actions without postconditions. This problem is in NP.*

Proof. We define the following algorithm:

procedure $existplan(s_0, A, \varphi_G)$
  (∃) choose an integer $k \in \{0, \ldots, |s_0|\}$
  (∃) choose actions $\alpha_1, \ldots, \alpha_k \in A$
  let $s = s_0$
  for $i = 1$ to k:
    if α_i is applicable in s **then** $s = s \otimes \alpha_i$ **else reject**
if $s \models \varphi_G$ **then accept else reject**

The algorithm tries to find a solution by choosing non deterministically the sequence of actions executed. It is important to notice that since the actions are separable and without postconditions, it is sufficient to consider a solution of at most $|s_0|$ actions, since each application of an action will either remove a world, an epistemic edge or do nothing. We then store the initial model into a state s and update this state by applying all actions α_i and failing if one is not applicable. Since actions are separable, then the product update is performed in polynomial time. Finally, we check that φ_G, which is a formula of $\mathcal{L}_K$, is satisfied in the epistemic state s. Since the model checking against epistemic logic is in PTIME [8] this concludes the proof.

□

Theorem 16. *We consider the plan existence problem under the constraint that A only contains separable actions. This problem is in PSPACE.*

Proof. We write a trivial algorithm searching for a solution:

procedure $existplan(s_0, A, \varphi_G)$
if $s_0 \models \varphi_G$ **then accept**
else :
  (∃) choose an action $\alpha \in A$
  if α is applicable in s_0 **then** $existplan(s \otimes \alpha, A, \varphi_G)$ **else reject**

[8] Epistemic logic is a fragment of CTL whose model checking is in PTIME [47].

It chooses an action and calls back to itself with the product update in input. It stops whenever the goal has been reached. Since the actions are separable, then $|s \otimes \alpha| \leq |s|$, so the space needed is polynomial. Therefore we have an algorithm in NPSPACE, which proves the PSPACE-membership since PSPACE = NPSPACE. □

Remark 2. In theory, the separability constraint is too strong. It is sufficient to say that for the input epistemic state s_0, for any actions $\alpha_1, \ldots, \alpha_n \in A$ and action $\alpha \in A$, in each world of $s_0 \otimes \alpha_1 \otimes \cdots \otimes \alpha_n$, at most one of the events of α is applicable. Indeed if this constraint is true, then the epistemic state will never increase in size, and so the algorithms presented before still work. Yet, this constraint is hard to check. That is why separability is an easier criterion. It is still possible to design an algorithm that applies the procedures of separable actions and stops if the current epistemic state increases in size. Similarly, if $s_0 \otimes \alpha_1 \otimes \ldots \alpha_n$ is never bigger than $P(|s_0|)$ for a fixed polynomial P then the Theorems 15 and 16 would still work (just replace $i \leq |s_0|$ by $i \leq P(|s_0|)$).

6. Related work

The undecidability of the plan existence problem for epistemic planning was originally established by Bolander and Andersen [11] (for the class $\mathcal{T}(1,0)$). The decidability of PlanEx-$\mathcal{T}(0,0)$ was proved by Yu et al. [49] (and non-elementary algorithms were provided by Aucher et al. [3] and Douéneau-Tabot et al. [21]). Furthermore, the PSPACE-complete result for PlanEx-$\mathcal{T}(0,-1)$ was proved by Charrier et al. [15], which improved the previous EXPSPACE upper bound of Bolander et al. [12].

To circumvent undecidability or such high complexities, other approaches have been investigated that revolve around syntactic compilation into classical planning [34, 27, 17]. The common point between these approaches is the restricted expressivity of either Kripke models or event models, so that the epistemic plan existence problem reduces to classical planning. For example, the setting of Muise et al. [34] permits event models with a single event with so-called *conditional effects* (non-deterministic postconditions), allowing for an exponential reduction to classical planning. In Cooper et al. [17], the authors also consider event models with a single event and conditional effects, the difference being that the Kripke model is implicitly formalized by valuations over a richer set of atomic propositions. This set contains new propositions of the form $Kw_a p$ for "Agent a knows whether p is true". In Kominis and Geffner[27], restricting to public announcements, public assignments and semi-private announcements yields a polynomial reduction to classical planning.[9]

Variants of the plan existence problem have also been considered that do not rely on DEL. For instance, one may use Alternating Temporal Logic (ATL) [1] extended to an imperfect information setting and enriched with knowledge modalities (AETL). ATL is itself an extension of computation-tree logic CTL [5] with dynamic modalities to express the existence of strategies. In this setting, one can express the existence of a strategy for a group of agents that ensures to reach a situation where some property φ holds, hence it can answer planning problems. Notice that the setting is very expressive and allows to express properties that go beyond planning issues. The model checker MCMAS [29] can solve AETL, but in a very restricted framework: the temporal dynamics (histories) rely on a finite (i.e., regular) model, only equivalence accessibility relations between states are allowed, and the knowledge semantics is *memoryless*, meaning that agents can only rely on their information about the current state and not about the whole history that led to it. This is opposed to the *perfect recall* knowledge semantics where agents remember all information along the history, as in the DEL setting. Our undecidability results entail the undecidability of the model checking problem for a perfect recall knowledge semantics, as independently proved by Dima and Tiplea [18]. Additionally, notice that whether the temporal dynamics resulting from the DEL structure is regular is still an open problem; it is regular for propositional event models, as explained in Section 4.1.1.

Incidentally, in the case of propositional actions, other works have established much stronger results that entail the decidability of the epistemic plan existence problem and its synthesis (as explained Section 4.1.1). For example, Maubert [32] and Bozzelli et al. [14] showed that model checking against CTL*K$_n$ is decidable over the class of DEL presentations; the automata-theoretic approach underlying this result allows to synthesize epistemic plans (action sequences), but it also allows to synthesize so-called *epistemic protocols*, in the sense of Aucher [3, Definition 6, page 4]. Notice that the more recent work by Douéneau-Tabot et al. [21] establishes the even stronger result that chain-MSO properties (a more expressive setting than CTL*K$_n$) can be model-checked over propositional DEL structures.

[9] This is expected since all these event models are separable.

More recently, Liberman and Rendsvig [28] have considered epistemic planning in the FODEL setting, that is, DEL extended with first-order (FO) logical features: under the assumption of propositional actions, the epistemic plan existence problem remains decidable, as long as the interpretation of the FODEL presentation carries a finite domain. Their elegant proof relies on the ability to finitely bound the number of different models (modulo bisimulation). An alternative proof would be to use our Theorem 7 that amounts to reasoning on automatic structures, since FODEL specifications interpreted over finite FO domains yield DEL presentations.

As already discussed in this section, the plan existence problem takes various forms and is still a very active area. The reader may refer to the recent report by Baral et al. [7] that provides a comprehensive survey on the many techniques that address this problem.

7. Conclusion

We surveyed complexity results of the plan existence problem based on dynamic epistemic logic (DEL). These results are summarized in Figure 25, where arrows denote an embedding of a class of actions into another.

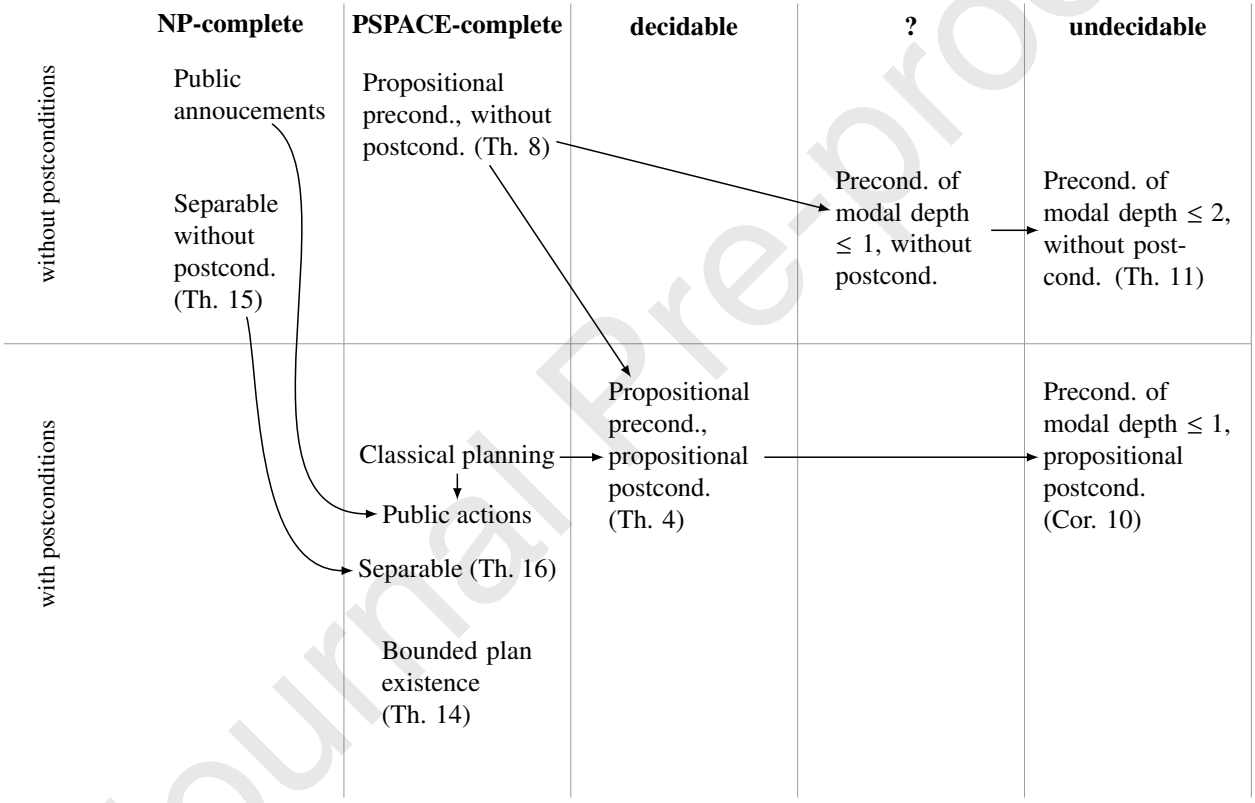


Figure 25: Overview of the plan existence problem complexity.

We investigated a classification of the (epistemic) plan existence problem based on bounding the modal depth of pre- and post-conditions of actions, which allowed us to embed many existing results in the literature. As a result, we obtained decidability only for propositional actions, but undecidability even for actions of modal depth 1. Nevertheless, the status of the plan existence problem remains open for actions with preconditions of modal depth 1 and no postconditions. However, we showed that the little addition of public announcements of modal depth 1 to the decidable case of propositional actions already yields undecidability. This last result reflects how tricky the open case is.

The established undecidability results may seem very disheartening, yet there is still hope for epistemic planning based on dynamic epistemic logic. Indeed, in addition to the case of propositional actions with no postconditions, we exhibited two more cases where the plan existence problem is decidable. The first is the bounded plan existence problem, where the length of the solution we seek is bounded by an input integer β. The second is the plan existence problem for separable actions, where the preconditions of any pair of events are logically inconsistent. For these cases, we obtained PSPACE-completeness complexity, just as for classical planning, which is very promising.

In the light of all these results, we can draw some conclusions. The hardness of epistemic planning lies in the combination of two factors: the growth of the epistemic state sizes and the conditionals for executing the events (*i.e.* the preconditions). When one of these factors is alleviated, the plan existence problem becomes decidable. Indeed, in the propositional action case, the model may grow arbitrarily, but the conditionals are sufficiently simple to be captured by regular models, namely automatic structures. On the other hand, for separable actions, the model does not grow at all, but the conditionals may be arbitrarily expressive. Actually, the concomitant presence of both factors entailing undecidability is highlighted by one of our results: gathering public announcements and propositional actions already makes the problem undecidable since propositional actions may make the epistemic state arbitrarily large while the public announcements introduce intricate conditionals. For future work, we advocate that the current undecidable classes appeal for a criteria to partition them into decidable classes.

The first avenue for future research is to formally study the necessary conditionals for executing events, in order to find properties such as "there exist a finite number of epistemic states modulo bisimulation" or "when precondition formulas belong to class ..., the DEL-structure is automatic". More generally, it is important to tame the classes of relational structures one can catch with DEL presentations. We have seen that those are automatic for propositional DEL presentations, and actually they fall into an even smaller class called *regular automatic trees* [19] with the very nice property that any chain Monadic Second Order Logic definable property can be checked. A deep investigation of which classes of structures are "DEL-presentable" is worth pursuing. In the same line, the recent work [28] opens a new horizon to consider FODEL (i.e. first-order DEL) presentations. While this recent work reveals decidability for propositional actions and finite first-order domains (and the proposed proof is ad hoc), we believe the proof via our Theorem 7 (see Section 6) can be generalized to subclasses of FODEL presentations with possibly infinite domain interpretations.

A second avenue is to study epistemic planning for meaningful cases, namely for those arising from practical applications, rather than in their full generality. Typically, for applications in security, it is relevant to study private communication event models, which currently do not appear as an entire class of our classification.

Acknowledgements. We would like to acknowledge Guillaume Aucher and the anonymous reviewers for significant contributions to this paper. The anonymous reviewers pointed out errors in the original submission as well as gave very helpful suggestions for improving the presentation of the paper. In particular one of the reviewers did an impressively thorough work surpassing all reasonable expectations. This resulted in significant improvements of the text. Guillaume Aucher played a crucial role in the early phases of the becoming of this paper by contributing to the conception and discussions concerning classifying epistemic planning classes in terms of model depth of pre- and post-conditions as well as the early brainstorming of ideas for the (un)decidability of different classes, in particular the open case of PlanEx-$\mathcal{T}(1,0)$, that is unfortunately still open at the time of writing.

References

[1] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.

[2] Guillaume Aucher and Thomas Bolander. Undecidability in epistemic planning. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)*, 2013.

[3] Guillaume Aucher, Bastien Maubert, and Sophie Pinchinat. Automata techniques for epistemic protocol synthesis. In *Proceedings 2nd International Workshop on Strategic Reasoning, SR 2014, Grenoble, France, April 5-6, 2014.*, pages 97–103, 2014.

[4] Guillaume Aucher and François Schwarzentruber. On the complexity of dynamic epistemic logic. *TARK 2013*, abs/1310.6406, 2013.

[5] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.

[6] Alexandru Baltag, Lawrence S. Moss, and Slawomir Solecki. The logic of public announcements and common knowledge and private suspicions. In *Proceedings of the 7th Conference on Theoretical Aspects of Rationality and Knowledge (TARK-98)*, pages 43–56, 1998.

[7] Chitta Baral, Thomas Bolander, Hans van Ditmarsch, and Sheila McIlrath. Epistemic planning (dagstuhl seminar 17231). In *Dagstuhl Reports*, volume 7(6). Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2017.

[8] Patrick Blackburn, Martin de Rijke, and Yde Venema. *Modal Logic*. Cambridge University Press, 2001.

[9] Patrick Blackburn, Johan van Benthem, and Frank Wolter. *Handbook of modal logic*, volume 3. Elsevier, 2006.

[10] Achim Blumensath and Erich Grädel. Automatic structures. In *Logic in Computer Science, 2000*, pages 51–62. IEEE, 2000.

[11] Thomas Bolander and Mikkel Birkegaard Andersen. Epistemic planning for single and multi-agent systems. *Journal of Applied Non-Classical Logics*, 21(1):9–34, 2011.

[12] Thomas Bolander, Martin Holm Jensen, and François Schwarzentruber. Complexity results in epistemic planning. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015*, pages 2791–2797, 2015.

[13] Blai Bonet and Hector Geffner. Planning with incomplete information as heuristic search in belief space. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems*, pages 52–61. AAAI Press, 2000.

[14] Laura Bozzelli, Bastien Maubert, and Sophie Pinchinat. Uniform strategies, rational relations and jumping automata. *Information and Computation*, 242:80–107, 2015.

[15] Tristan Charrier, Bastien Maubert, and François Schwarzentruber. On the impact of modal depth in epistemic planning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016*, pages 1030–1036, 2016.

[16] Sébastien Lê Cong, Sophie Pinchinat, and François Schwarzentruber. Small undecidable problems in epistemic planning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pages 4780–4786, 2018.

[17] Martin C. Cooper, Andreas Herzig, Faustine Maffre, Frédéric Maris, and Pierre Régnier. A simple account of multi-agent epistemic planning. In *Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI 2016)*, pages 193–201, 2016.

[18] Catalin Dima and Ferucio Laurentiu Tiplea. Model-checking ATL under imperfect information and perfect recall semantics is undecidable. *CoRR*, abs/1102.4225, 2011.

[19] Gaëtan Douéneau-Tabot, Sophie Pinchinat, and François Schwarzentruber. Chain-monadic second order logic over regular automatic trees and epistemic planning synthesis. In *Advances in Modal Logic 7*, 2018.

[20] Gaëtan Douéneau-Tabot. Propriétés régulières des arbres, 2016.

[21] Gaëtan Douéneau-Tabot, Sophie Pinchinat, and François Schwarzentruber. Chain-monadic second order logic over regular automatic trees and epistemic planning synthesis. In *Proceedings of AiML2018*, 2018.

[22] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning about Knowledge*. MIT Press, 1995.

[23] Richard Fikes and Nils J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. In *Proceedings of the 2nd International Joint Conference on Artificial Intelligence*, pages 608–620, 1971.

[24] Malte Helmert. *Understanding planning tasks: Domain complexity and heuristic decomposition*, volume 4929. Springer, 2008.

[25] Jaakko Hintikka. *Knowledge and Belief: An Introduction to the Logic of the Two Notions*. Cornell University Press, 1962.

[26] Martin Holm Jensen. *Epistemic and doxastic planning*. PhD thesis, Technical University of Denmark (DTU), 2014.

[27] Filippos Kominis and Hector Geffner. Beliefs in multiagent planning: From one agent to many. In *Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling, ICAPS 2015*, pages 147–155, 2015.

[28] Andrés Occhipinti Liberman and Rasmus Kræmmer Rendsvig. Decidability results in first-order epistemic planning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI*, 2020.

[29] Alessio Lomuscio, Hongyang Qu, and Franco Raimondi. MCMAS: an open-source model checker for the verification of multi-agent systems. *Software Tools for Technology Transfer*, 19(1):9–30, 2017.

[30] Benedikt Löwe, Eric Pacuit, and Andreas Witzel. DEL planning and some tractable cases. In *Logic, Rationality, and Interaction - Third International Workshop, LORI 2011*, pages 179–192, 2011.

[31] João Marques-Silva. Computing with SAT oracles: Past, present and future. In *14th Conference on Computability in Europe, CiE 2018*, pages 264–276, 2018.

[32] Bastien Maubert. *Logical foundations of games with imperfect information: Uniform strategies*. PhD thesis, Université de Rennes 1, 2014.

[33] Marvin L. Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1967.

[34] Christian J. Muise, Vaishak Belle, Paolo Felli, Sheila A. McIlraith, Tim Miller, Adrian R. Pearce, and Liz Sonenberg. Planning over multi-agent epistemic states: A classical planning approach. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 3327–3334, 2015.

[35] Christos H Papadimitriou. *Computational complexity*. John Wiley and Sons Ltd., 2003.

[36] David Premack and Guy Woodruff. Does the chimpanzee have a theory of mind? *Behavioral and Brain Sciences*, 1(4):515–526, 1978.

[37] Jussi Rintanen. Complexity of planning with partial observability. In *Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling*, pages 345–354, 2004.

[38] Sasha Rubin. Automata presenting structures: A survey of the finite string case. *Bulletin of Symbolic Logic*, 14(2):169–209, 2008.

[39] Tomasz Sadzik. Exploring the Iterated Update Universe. ILLC Publications PP-2006-26, 2006.

[40] Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970.

[41] François Schwarzentruber. Hintikka's world: Agents with higher-order knowledge. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pages 5859–5861, 2018.

[42] Alvy Ray Smith. Simple computation-universal cellular spaces and self-reproduction. In *9th Annual Symposium on Switching and Automata Theory*, pages 269–277, 1968.

[43] Hudson Turner. Polynomial-length planning spans the polynomial hierarchy. In *Logics in Artificial Intelligence, European Conference, JELIA 2002*, pages 111–124, 2002.

[44] Johan van Benthem, Jelle Gerbrandy, Tomohiro Hoshi, and Eric Pacuit. Merging frameworks for interaction. *Journal of Philosophical Logic*, 38(5):491–526, 2009.

[45] Hans van Ditmarsch and Barteld Kooi. *One Hundred Prisoners and a Light Bulb*. Springer, 2015.

[46] Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Kooi. *Dynamic Epistemic Logic*. Springer, Dordecht, 2008.

[47] Moshe Y. Vardi. Automatic verification of probabilistic concurrent finite state programs. In *26th Annual Symposium on Foundations of Computer Science*, pages 327–338. IEEE, 1985.

[48] Stephen Wolfram. *A new kind of science*. Wolfram-Media, 2002.

[49] Quan Yu, Ximing Wen, and Yongmei Liu. Multi-agent epistemic explanatory diagnosis via reasoning about actions. In *Proceedings of the International Joint Conference on Artificial Intelligence IJCAI*, 2013.

Appendix

In subsequent proofs, we need a criteria for equivalence between epistemic states. This criteria is *bisimulation* as defined in [8].

Definition 15 (Bisimulation)**.** Let $P \subseteq AP$ and $(\mathcal{M}, w_0)$, $(\mathcal{M}', w_0')$ be two epistemic states with $\mathcal{M} = (\mathsf{W}, (\rightarrow_a)_{a\in Ag}, \mathsf{V})$ and $\mathcal{M}' = (\mathsf{W}', (\rightarrow'_a)_{a\in Ag}, \mathsf{V}')$. We say that $(\mathcal{M}, w_0)$, $(\mathcal{M}', w_0')$ are *P-bisimilar* if there exists a relation $B \subseteq \mathsf{W} \times \mathsf{W}'$ such that $(w_0, w_0') \in B$ and for all $(w, w') \in B$:

- P-conservation: for all $p \in P$, $p \in \mathsf{V}(w)$ if and only if $p \in \mathsf{V}'(w')$;
- Zig: if $w \rightarrow_a u$ then there exists $u' \in \mathsf{W}'$ such that $w' \rightarrow'_a u'$ and $(u, u') \in B$;
- Zag: if $w' \rightarrow'_a u'$ then there exists $u \in \mathsf{W}$ such that $w \rightarrow_a u$ and $(u, u') \in B$.

Appendix A. Proof of Theorem 2

In this proof we need the definition of the set of subformulas of a formula φ. It is noted $SF(\varphi)$, and is defined by structural induction on φ: $SF(p) = \{p\}$; $SF(\neg\varphi) = \{\neg\varphi\} \cup SF(\varphi)$; $SF(\varphi_1 \vee \varphi_2) = \{\varphi_1 \vee \varphi_2\} \cup SF(\varphi_1) \cup SF(\varphi_2)$; $SF(K_a\varphi) = \{K_a\varphi\} \cup SF(\varphi)$.

Let $T = (s_0, A, \varphi_g)$ be a planning task in $\mathcal{T}(m, n)$ and let $P \subseteq AP$ be the set of atomic propositions appearing in T. We define the following set P':

$$P' = P \cup \{p_\psi \mid \psi \in SF(\varphi),\ \varphi \text{ appears in some action } \alpha \in A,\ md(\psi) \geq 1\} \cup \{p_1, \ldots, p_{\max(m,n)}\} \cup \{p_\alpha \mid \alpha \in A\} \cup \{p_{exec}\}$$

For any formula φ, we define a new formula $tr(\varphi)$ of modal depth at most one as follows: for formulas φ that already are of modal depth at most one, $tr(\varphi) = \varphi$. For formulas φ of modal depth at least two, $tr(\varphi)$ is φ where its subformulas ψ of modal depth $md(\varphi) - 1$ have been replaced by p_ψ: $tr(p) = p$; $tr(\top) = \top$; $tr(\neg\varphi) = \neg tr(\varphi)$; $tr(\varphi_1 \vee \varphi_2) = tr(\varphi_1) \vee tr(\varphi_2)$; $tr(K_a\psi) = K_a p_\psi$.

We construct a planning task $T' = (s_0', A', \varphi_g')$ from T as follows: $s_0' = s_0$, $\varphi_g' = \varphi_g \wedge \neg p_{exec} \wedge \bigwedge_{i=1}^{\max(m,n)} \neg p_i$ and

$$A' = \bigcup_{\alpha\in A} \{\alpha_{choose}, \alpha_1, \ldots, \alpha_{\max(m,n)}, \alpha'\}$$

where the new actions for simulating $\alpha = (\mathcal{E}, e_0)$ with $\mathcal{E} = (\mathsf{E}, (\rightarrow_a)_{a\in Ag}, \mathrm{pre}, \mathrm{post})$ are the following.

- $\alpha_{choose} = (\mathcal{E}_{choose}, e_{choose})$, $\mathcal{E}_{choose} = (\{e_{choose}\}, (\{(e_{choose}, e_{choose})\})_{a\in Ag}, \mathrm{pre}_{choose}, \mathrm{post}_{choose})$ with $\mathrm{pre}_{choose}(e_{choose}) = \neg p_{exec} \wedge \bigwedge_{i=1}^{\max(m,n)} \neg p_i$ and $\mathrm{post}_{choose}(e_{choose})(p_1) = \top$; $\mathrm{post}_{choose}(e_{choose})(p_\alpha) = \top$.
- $\alpha_k = (\mathcal{E}_k, e_k)$, $\mathcal{E}_k = (\{e_k\}, (\{(e_k, e_k)\})_{a\in Ag}, \mathrm{pre}_k, \mathrm{post}_k)$ with $\mathrm{pre}_k(e_k) = p_k \wedge p_\alpha$ and $\mathrm{post}_k(e_k)(p_\varphi) = tr(\varphi)$ for any formula φ of modal depth k such that $p_\varphi \in P'$; $\mathrm{post}_k(e_k)(p_k) = \bot$; if $k < \max(m, n)$ then $\mathrm{post}_k(e_k)(p_{k+1}) = \top$ else $\mathrm{post}_k(e_k)(p_{exec}) = \top$.
- $\alpha' = (\mathcal{E}', e_0')$, $\mathcal{E}' = (\mathsf{E}', (\rightarrow'_a)_{a\in Ag}, \mathrm{pre}', \mathrm{post}')$ with $\mathsf{E}' = \{e' \mid e' \in \mathsf{E}\}$. $\rightarrow'_a = \{(e', f') \mid e \rightarrow_a f\}$ for all $a \in Ag$. $\mathrm{pre}'(e') = p_{exec} \wedge p_\alpha \wedge tr(\mathrm{pre}(e))$ for all $e' \in \mathsf{E}'$. $\mathrm{post}'(e')(p) = tr(\mathrm{post}(e)(p))$ for all $e' \in \mathsf{E}'$ and $p \in P$; $\mathrm{post}'(e')(p) = \bot$ for all $e' \in \mathsf{E}'$ and $p \in P' \setminus P$.

Lemma 17. *For any epistemic state s and action α over a set of atomic propositions P, the epistemic states $s \otimes \alpha$ and $s \otimes \alpha_{choose} \otimes \bigotimes_{k=1}^{\max(m,n)} \alpha_k \otimes \alpha'$ are P-bisimilar.*

Proof. Let $s = (\mathcal{M}, w_0)$ with $\mathcal{M} = (\mathsf{W}, (\rightarrow_a)_{a\in Ag}, \mathsf{V})$ and $\alpha = (\mathcal{E}, e_0)$ with $\mathcal{E} = (\mathsf{E}, (\rightarrow_a)_{a\in Ag}, \mathrm{pre}, \mathrm{post})$. The bisimulation is $B = \{((w, e), (w, e_{choose}, e_1, \ldots, e_{\max(m,n)}, e') \mid w \in \mathsf{W}, e \in \mathsf{E}\}$. The Zig and Zag properties are easy to prove, the difficult part is to prove that $(\mathcal{M}, w) \models \varphi$ if and only if $(\mathcal{M}, w) \otimes \alpha_{choose} \otimes \bigotimes_{k=1}^{\max(m,n)} \alpha_k \models tr(\varphi)$ with φ any formula appearing in a precondition or a postcondition in α. Indeed, if such a property is proven, then:

- B is well defined because $(w, e_{choose}, e_1, \ldots, e_{\max(m,n)}, e')$ passed all the preconditions if and only if w passed the precondition of e (because the only problematic precondition is $\text{pre}'(e') = tr(\text{pre}(e))$).
- The P-conservation is direct because:

$$\begin{array}{ll} p \in \mathsf{V}(w,e) & \text{if and only if } (\mathcal{M},w) \models \text{post}(e)(p) \\ & \text{if and only if } (\mathcal{M},w) \otimes \alpha_{choose} \otimes \bigotimes_{k=1}^{\max(m,n)} \alpha_k \models tr(\text{post}(e)(p)) \\ & \text{if and only } (\mathcal{M},w) \otimes \alpha_{choose} \otimes \bigotimes_{k=1}^{\max(m,n)} \alpha_k \models \text{post}'(e')(p) \\ & \text{if and only if } p \in \mathsf{V}(w, e_{choose}, e_1, \ldots, e_{\max(m,n)}). \end{array}$$

The property $(\mathcal{M},w) \otimes \alpha_{choose} \otimes \bigotimes_{k=1}^{\max(m,n)} \alpha_k \models tr(\varphi)$ is a consequence of the following lemma.

Lemma 18. *For any $i \in \{1, \ldots, \max(m,n)\}$, for any formula φ of modal depth at most i and any Kripke model $\mathcal{M}$ with its relevant propositions on P, $(\mathcal{M},w) \models \psi$ if and only if $(\mathcal{M},w) \otimes \alpha_{choose} \otimes \bigotimes_{k=1}^{i} \alpha_k \models tr(\psi)$.*

By induction on i.

- For $i = 1$, we have $tr(\psi) = \psi$ and $(\mathcal{M},w) \otimes \alpha_{choose} \otimes \bigotimes_{k=1}^{i} \alpha_k = (\mathcal{M},w) \otimes \alpha_{choose} \otimes \alpha_1$. Since α_{choose} and α_1 only contain one event, do not remove any world from $\mathcal{M}$ and do not modify any proposition from P, we directly obtain that $(\mathcal{M},w) \models \psi$ if and only if $(\mathcal{M},w) \otimes \alpha_{choose} \otimes \alpha_1 \models tr(\psi)$.
- For $i > 1$: if the property is true for $i-1$ then in $(\mathcal{M},w) \otimes \alpha_{choose} \otimes \bigotimes_{k=1}^{i} \alpha_k$, any atomic proposition p_χ with χ of modal depth $i-1$ has now the correct value of χ in each world w. Therefore for any formula ψ of modal depth i, we indeed have $(\mathcal{M},w) \models \psi$ if and only if $(\mathcal{M},w) \otimes \alpha_{choose} \otimes \bigotimes_{k=1}^{i} \alpha_k \models tr(\psi)$.

□

For any action α we define the abbreviation $\pi_\alpha = \alpha_{choose} \otimes \bigotimes_{k=1}^{\max(m,n)} \alpha_k \otimes \alpha'$. Then we have proved that $\mathcal{M} \otimes \alpha_1 \otimes \cdots \otimes \alpha_k$ is P-bisimilar to $\mathcal{M} \otimes \pi_{\alpha_1} \otimes \cdots \otimes \pi_{\alpha_k}$, so the goal formula φ_g is either true in both of false in both. Since the rest of the conjunction in φ'_g is necessarily true in $\mathcal{M} \otimes \pi_{\alpha_1} \otimes \cdots \otimes \pi_{\alpha_k}$, we have proved that φ_g is true in $\mathcal{M} \otimes \alpha_1 \otimes \cdots \otimes \alpha_k$ if and only if φ'_g is true in $\mathcal{M} \otimes \pi_{\alpha_1} \otimes \cdots \otimes \pi_{\alpha_k}$.

We conclude the proof by highlighting that necessarily the plans are of the form $\pi_{\alpha_1}, \ldots, \pi_{\alpha_k}$. Indeed, each π_{α_j} cannot be cut down because of the proposition p_{α_j}, and φ'_g is necessarily false if evaluated before π_{α_j} is finished because all propositions p_i and p_{exec} must be false.

Appendix B. Proof of Theorem 3

Let $T = (s_0, A, \varphi_g)$ be a planning task in $\mathcal{T}(m,n)$ over a set of atomic propositions P, with $s_0 = (\mathcal{M}, w_0)$, $\mathcal{M} = (\mathsf{W}, (\rightarrow_a)_{a \in Ag}, \mathsf{V})$. Let $P' = P \cup P_{post} \cup \{p_\alpha \mid \alpha \in A\}$ with $P_{post} = \{p_{\text{post}(e)(p)} \mid e \text{ appears in some action } \alpha \in A,\ p \in P\}$. We first detail an exponential reduction and then explain how to extract a polynomial reduction. From T, we construct a planning task $T' = (s'_0, A', \varphi'_g)$ in $\mathcal{T}(\max\{m,n\}, 0)$ over P' by letting $s'_0 = s_0$, $\varphi'_g = \varphi_g \wedge \bigwedge_{\alpha \in A} \neg p_\alpha$ and

$$A' = \bigcup_{\alpha \in A} \{\alpha^v_{assign} \mid v \in 2^{P_{post}}\} \cup \{\alpha'\}$$

where the new actions for simulating $\alpha = (\mathcal{E}, e_0)$ with $\mathcal{E} = (\mathsf{E}, (\rightarrow_a)_{a \in Ag}, \text{pre}, \text{post})$ are the following.

- $\alpha^v_{assign} = (\mathcal{E}_{assign}, e_v)$, $\mathcal{E}_{assign} = (\mathsf{E}_{assign}, (\rightarrow_{a,assign})_{a \in Ag}, \text{pre}_{assign}, \text{post}_{assign})$ with
  - $\mathsf{E}_{assign} = \{e_v \mid v \in 2^{P_{post}}\}$;
  - $\rightarrow_{a,assign} = \mathsf{E}_{assign} \times \mathsf{E}_{assign}$;
  - $\text{pre}_{assign}(e_v) = \bigwedge_{\alpha \in A} \neg p_\alpha \wedge \bigwedge_{\text{post}(e)(p) \text{ in } \alpha \text{ and } v} \text{post}(e)(p) \wedge \bigwedge_{\text{post}(e)(p) \text{ in } \alpha \text{ and not in } v} \neg\, \text{post}(e)(p)$;
  - $\text{post}_{assign}(e_v)(p_{\text{post}(e)(p)}) = \begin{cases} \top \text{ if } \text{post}(e)(p) \in v \\ \bot \text{ otherwise} \end{cases}$; $\text{post}_{assign}(e_v)(p_\alpha) = \top$.

$\tau(p)(x)$	:=	$p(x)$	for each $p \in AP$
$\tau(\neg\varphi)(x)$	:=	$\neg\tau(\varphi)(x)$	
$\tau(\psi \wedge \chi)(x)$	:=	$\tau(\psi)(x) \wedge \tau(\chi)(x)$	
$\tau(K_a\varphi)(x)$	:=	$\forall y(R_a(x,y) \rightarrow \tau(\varphi)(y))$	

Table C.1: Translation of $\mathcal{L}_{\mathrm{K}}$ into FO.

- $\alpha' = (\alpha, e'_0)$, $\mathcal{E}' = (\mathsf{E}', (\rightarrow'_a)_{a\in Ag}, \mathrm{pre}', \mathrm{post}')$ with:
  - $\mathsf{E}' = \{e' \mid e \in \mathsf{E}\}$;
  - $\rightarrow'_a = \{(e', f') \mid e \rightarrow_a f\}$ for all $a \in Ag$;
  - $\mathrm{pre}'(e') = p_\alpha \wedge \mathrm{pre}(e)$ for all $e' \in \mathsf{E}'$;
  - $\mathrm{post}'(e')(p) = p_{\mathrm{post}(e)(p)}$ for all $e \in \mathsf{E}$ and $p \in P$; $\mathrm{post}'(e')(p) = \bot$ for all $p \in P' \setminus P$.

The correctness proof is direct for this construction. Indeed, let α^v_{assign} be the only applicable action in $(\mathcal{M}, w)$. Then in each world of $\mathcal{M}$, only one event of α^v_{assign} is executed and tags the propositions $p_{\mathrm{post}(e)(p)}$ to the correct value of $\mathrm{post}(e)(p)$. Therefore $(\mathcal{M}, w)$ is exactly the same as $(\mathcal{M}, w)\otimes\alpha^v_{assign}$ with $p_{\mathrm{post}(e)(p)}$ tagged with the correct value and p_α true in every world. Therefore, $(\mathcal{M}, w)\otimes\alpha$ and $(\mathcal{M}, w)\otimes\alpha^v_{assign}\otimes\alpha'$ are P-bisimilar. Therefore $(\mathcal{M}, w)\otimes\alpha_1\otimes\cdots\otimes\alpha_k \models \varphi_g$ if and only if $(\mathcal{M}, w) \otimes (\alpha_1)_{assign} \otimes \alpha'_1 \otimes \cdots \otimes (\alpha_k)_{assign} \otimes \alpha'_k \models \varphi'_g$, since the rest of the formula φ'_g forbids the plans to end with some $(\alpha_i)_{assign}$.

To obtain a polynomial reduction, we divide the action α^v_{assign} into actions $\alpha_{\mathrm{post}(e)(p)}$ having two events and just assigning the value of $\mathrm{post}(e)(p)$. Since the actions α^v_{assign} differ only in the designated event, we introduce two actions $\alpha_{\mathrm{post}(e)(p)}$ for each $\mathrm{post}(e)(p)$. By also introducing new propositions to impose that all $\alpha_{\mathrm{post}(e)(p)}$ are executed in order, we thus can simulate the set of α^v_{assign} with a polynomial number of actions of size two, which makes the reduction polynomial.

Appendix C. Epistemic logic can be translated into FO from page 19

Recall that epistemic logic is interpreted over Kripke models (see Definition 2) that are particular relational structures: a Kripke model $\mathcal{M} = (\mathsf{W}, (\rightarrow_a)_{a\in Ag}, \mathsf{V})$ is the relational structure whose domain is W and whose relations are the binary relations $\rightarrow_a \subseteq \mathsf{W}\times\mathsf{W}$, one for each $a \in Ag$, and the unary relations $\mathsf{V}(p) \subseteq \mathsf{W}$, one for each $p \in AP$. For readability of the FO-formulas, the binary relations $\rightarrow_a$ and the unary relations $\mathsf{V}(p)$ will instead be written R_a and p.

We now define a translation $\tau : \mathcal{L}_{\mathrm{K}} \rightarrow \mathsf{FO}$ that maps a formula φ onto a first-order formula with a single variable, hence written $\tau(\varphi)(x)$ to emphasize this free variable as x. Mapping τ is designed so that given an epistemic state $(\mathcal{M}, w)$ and a formula $\varphi \in \mathcal{L}_{\mathrm{K}}$, $(\mathcal{M}, w) \models \varphi$ if, and only if, $\mathcal{M}, [x \mapsto w] \models \tau(\varphi)(x)$.

The translation is defined in Table C.1 obtained by induction over the formula φ. It expends the semantics of the knowledge operator K_a as a universal quantification over the R_a-neighborhood of a world: for example, the formula $K_a p \wedge \hat{K}_b q$ translates into $\Psi(x) := \forall y(R_a(x,y) \rightarrow p(y)) \wedge \exists z(R_b(x,z) \wedge q(z))$.

For the correctness of translation τ, known as *the standard translation* in the literature, the reader may refer to [9, Section 2.2].

Appendix D. Proof of Theorem 12

For pedagogical reasons, we first discuss an exponential reduction from PlanEx-$\mathcal{T}(1,0)$ to PlanEx-$\mathcal{T}((0,0) \cup (1,-1))$. The trick to make it polynomial is the same as for reduction given in the proof of Theorem 3.

Appendix D.1. Overall idea

Let us illustrate our reduction with the action α from Figure D.26. For the next paragraph we ignore the role of $exec_\alpha, e_{tagAssign}, e_{tagLost}, flagAssigned, flagLost$. Action α is simulated by the execution of the three actions $\alpha_{assign} \Rightarrow \alpha_{check} \Rightarrow \alpha'$, executed in that order:

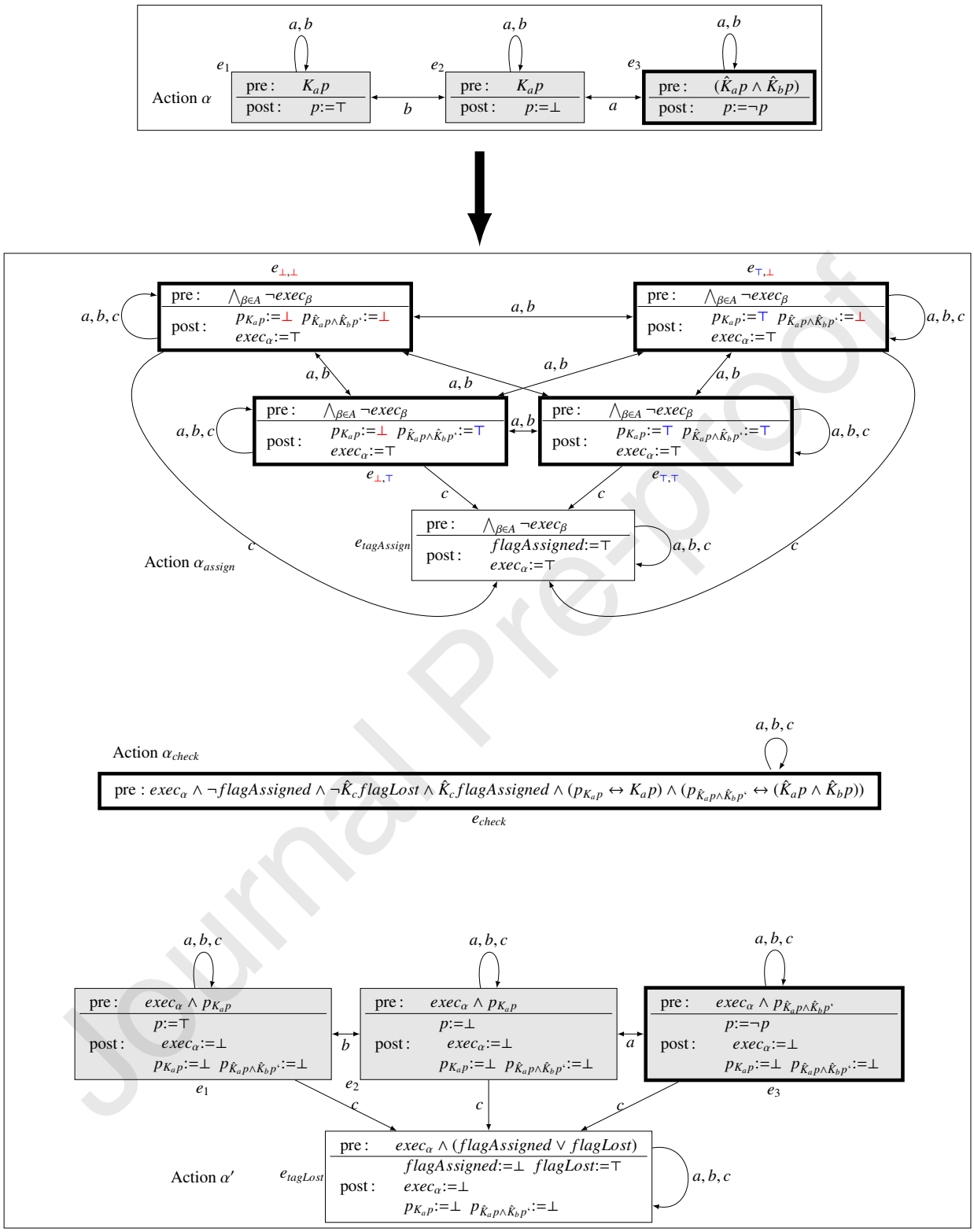


Figure D.26: Example of an action α in $\mathcal{T}(1,0)$ that is simulated by actions α_{assign}, α_{check} and α' in $\mathcal{T}(0,0) \cup \mathcal{T}(1,-1)$).

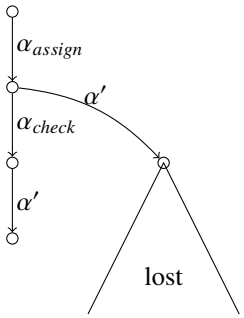


Figure D.27: Possible orders of the actions α_{assign}, α_{check} and α'.

1. α_{assign} is a propositional action that copies all worlds augmented with all possible assignments ν on new atomic propositions $p_{K_a p}$ and $p_{\hat{K}_a p \wedge \hat{K}_b p}$, where $\varphi_1 = K_a p$ and $\varphi_2 = \hat{K}_a p \wedge \hat{K}_b p$ are the (epistemic) preconditions appearing in α. These new propositional variables serve as placeholders whose truth values will be compared with the actual truth value of formulas φ_1 and φ_2 with action α_{check}. Since we do not know in advance which event is executable in the designated world, we create copies of α_{assign} where each possible assignment on $p_{\varphi_1}, \ldots, p_{\varphi_n}$ is designated. That is why we have several designated events in α_{assign}.

2. α_{check} is a public announcement that keeps all worlds in which the value of p_ψ is equal to the truth value of formula ψ.

3. α' is a propositional action consisting of a copy of α where each epistemic precondition ψ is replaced by p_ψ.

Now, imposing the order of execution $\alpha_{assign} \Rightarrow \alpha_{check} \Rightarrow \alpha'$ is tricky.

1. In order to have α_{assign} as the first action being executed for simulating an action α, we introduce a propositional variable $exec_\alpha$ that remains true during all the simulation of action α. Thus actions α_{check}, α' are for sure executed after α_{assign}, and no actions $\beta_{check}, \beta_{assign}$ with $\beta \neq \alpha$ can be executed after α_{assign}.

2. Now α_{check} is a public announcement: it cannot modify another atomic proposition to impose that α_{check} is executed between α_{assign} and α'. We then rely on the method of manipulating a *flag* in an extra possible world for a new agent c. We add to action α_{assign} a new event $e_{tagAssign}$ that will add a successor world for agent c where a certain proposition $flagAssigned$ is true. The execution of α_{check} requires $\hat{K}_a flagAssigned$ to be true, that is, requires that α_{assign} was executed just before. Also, as α_{check} contains $\neg flagAssigned$ in its precondition, its execution will remove all $flagAssigned$-worlds. Thus, as action α_{check} requires formula $\hat{K}_a flagAssigned$ to be true, action α_{check} cannot be perfored two times in a row.

3. Second, we add to α' a new event $e_{tagLost}$ that tags $flagAssigned$-worlds with $flagLost$ and keeps $flagLost$-worlds. The presence of $flagLost$-worlds mean that the simulation is incorrect, it happens when α' is executed just after α_{assign}. The presence of some $flagLost$-worlds implies that there will always be some $flagLost$-worlds: the whole simulation is incorrect forever.

Figure D.27 shows the possible orders of the actions. The goal formula is $\varphi'_g = \varphi_g \wedge \bigwedge_{\beta \in A} \neg exec_\beta \wedge \neg \hat{K}_c flagLost$:

- φ_g the previous goal should be satisfied;
- $\bigwedge_{\beta \in A} \neg exec_\beta$: the goal should be evaluated between simulations of original actions;
- $\neg \hat{K}_c flagLost$: the simulation should not be incorrect.

Appendix D.2. Formal definition of the reduction

Let $T = (s_0, A, \varphi_g)$ be a planning task in $\mathcal{T}(1,0)$ over a set of atomic propositions P, with $s_0 = (\mathcal{M}, w_0)$, $\mathcal{M} = (\mathsf{W}, (\to_a)_{a\in Ag}, \mathsf{V})$. Let $P' = P \cup \bigcup_{\alpha\in A} P_\alpha \cup \{flagAssigned, flagLost\} \cup \{exec_\alpha \mid \alpha \in A\}$ with $P_\alpha = \{p_\varphi \mid \varphi \text{ appears in } \alpha\}$.

We define a new planning task $T' = (s_0', A', \varphi_g')$ in $\mathcal{T}((0,0) \cup (1,-1))$ on P', where the actions of $\mathcal{T}(1,-1)$ are public announcements. First, s_0' is the state s_0. We introduce an new agent c with $\to_c = \{(w,w) \mid w \in \mathsf{W}\}$. Second $\varphi_g' = \varphi_g \wedge \bigwedge_{\alpha\in A} \neg exec_\alpha \wedge \neg \hat{K}_c flagLost$. Finally, the set actions is $A' = \bigcup_{\alpha\in A} \left(\{\alpha^v_{assign} \mid v \in 2^{P_\alpha}\} \cup \{\alpha_{check}, \alpha'\}\right)$, that are defined as follows, provided $\alpha = (\mathcal{E}, e_0)$.

- $\alpha^v_{assign} = (\mathcal{E}_{assign}, e_v)$, $\mathcal{E}_{assign} = (\mathsf{E}_{assign}, (\to_{a,assign})_{a\in Ag}, \mathrm{pre}_{assign}, \mathrm{post}_{assign})$ with
  - $\mathsf{E}_{assign} = \{e_v \mid v \in 2^{P_\alpha}\} \cup \{e_{tagAssign}\}$;
  - $\to_{a,assign} = \{(e_v, f_{v'}) \mid v, v' \in 2^{P_\alpha}\} \cup \left\{(e_{tagAssign}, e_{tagAssign})\right\}$ for all agents a different from c;
    $\to_{c,assign} = \{(e_v, f_v) \mid v \in 2^{P_\alpha}\} \cup \left\{(e_v, e_{tagAssign}) \mid v \in 2^{P_\alpha}\right\} \cup \left\{(e_{tagAssign}, e_{tagAssign})\right\}$;
  - $\mathrm{pre}_{assign}(e) = \bigwedge_{\beta\in A} \neg exec_\beta$ for all $e \in \mathsf{E}_{assign}$;
  - $\mathrm{post}_{assign}(e)(exec_\alpha) = \top$ for all $e \in \mathsf{E}_{assign}$,
    $\mathrm{post}_{assign}(e_v)(p_\varphi) = \begin{cases} \top \text{ if } p_\varphi \in v \\ \bot \text{ otherwise} \end{cases}$, $\mathrm{post}_{assign}(e_{tagAssign})(flagAssigned) = \top$.
- $\alpha_{check} = (\mathcal{E}_{check}, e_{check})$, $\mathcal{E}_{check} = (\{e_{check}\}, (\{(e_{check}, e_{check})\})_{a\in Ag}, \mathrm{pre}_{check})$ is without postconditions with

$$\mathrm{pre}_{check}(e_{check}) = exec_\alpha \wedge \neg flagAssigned \wedge \neg \hat{K}_c flagLost \wedge \hat{K}_c flagAssigned \wedge \bigwedge_{\varphi \text{ appearing in } \alpha} (p_\varphi \leftrightarrow \varphi).$$

- $\alpha' = (\mathcal{E}', e_0')$, $\mathcal{E}' = (\mathsf{E}', (\to_a')_{a\in Ag}, \mathrm{pre}', \mathrm{post}')$ with
  - $\mathsf{E}' = \{e' \mid e \in \mathsf{E}_\alpha\} \cup \{e_{tagLost}\}$
  - $\to_a' = \{(e', f') \mid e \to_a f\} \cup \{(e_{tagLost}, e_{tagLost})\}$ for all agents a different from c;
    $\to_c' = \{(e', e'), (e', e_{tagLost}) \mid e \in \mathsf{E}_\alpha\} \cup \{(e_{tagLost}, e_{tagLost})\}$,
  - $\mathrm{pre}'(e') = p_{\mathrm{pre}(e)} \wedge exec_\alpha$, $\mathrm{pre}'(e_{tagLost}) = exec_\alpha \wedge (flagAssigned \vee flagLost)$.
  - $\mathrm{post}'(f')(p) = p_{\mathrm{post}(e)(p)}$ for $p \in P$ and $f' \neq e_{tagLost}$,
    $\mathrm{post}'(e_{tagLost})(flagAssigned) = \bot$, $\mathrm{post}'(e_{tagLost})(flagLost) = \top$,
    $\mathrm{post}'(e')(exec_\alpha) = \bot$ for all events $e' \in \mathsf{E}'$.

We now prove the correctness of this construction. We begin by proving the following lemma.

Lemma 19. *The plans for T' are of the form $(\alpha_1)^{v_1}_{assign}, (\alpha_1)_{check}, \alpha_1', \ldots, (\alpha_n)^{v_n}_{assign}, (\alpha_n)_{check}, \alpha_n'$.*

To prove the lemma, we introduce the following property *Pr* on epistemic states: "In the designated world, $\bigwedge_{\beta\in A} \neg exec_\beta \wedge \neg \hat{K}_c flagLost$ is true".

Notice that φ_g' can be only true in models where *Pr* is true. Let us see the potential actions executable from a epistemic state where *Pr* is true. First, only the actions α_{assign} are executable for some action $\alpha \in A$, since *exec* is false. Then, there are two possibilities:

- We can execute α_{check}. In this case the formula $\neg \hat{K}_c flagLost$ remains true since α_{assign} only removes worlds. We cannot execute α_{check} a second time since all $flagAssigned$-worlds are removed, so the formula $\hat{K}_c flagAssigned$ is now false everywhere. After executing α_{check}, we can only execute α'. In that case all c-successors of the designated world are not tagged by $flagAssigned$ (because α_{check} removed them) or $flagLost$ (because $\neg \hat{K}_c flagLost$ is true). Therefore $e_{tagLost}$ is not applicable in these worlds, so no c-successor of the designated world is tagged by $flagLost$ after executing α'. After executing α', the formula $\bigwedge_{\beta\in A} \neg exec_\beta$ is also true in the designated world, therefore *Pr* holds.

- We can execute directly α', but then the formula $\neg \hat{K}_c flagLost$ will be false in the new epistemic state, therefore Pr is not true. The goal formula φ'_g is then false in this model, and it will never be true since we cannot assign $flagLost$ to false by any action and if we remove worlds tagged by $flagLost$ with α_{check}, we also remove all worlds at the same time because of the $\neg \hat{K}_c flagLost$ constraint.

Therefore, in models where Pr is true, the only way to arrive to a goal formula is to reach another model where Pr is true, thus executing a sequence of actions of the form $(\alpha_1)^{v_1}_{assign}, (\alpha_1)_{check}, \alpha_1', \ldots, (\alpha_n)^{v_n}_{assign}, (\alpha_n)_{check}, \alpha_n'$. Since s_0 is also a model where Pr is true, this concludes the proof of the lemma.

We now prove the following lemma.

Lemma 20. *$\alpha_1, \ldots, \alpha_n$ is a plan for T if and only if $(\alpha_1)^{v_1}_{assign}, (\alpha_1)_{check}, \alpha_1', \ldots, (\alpha_n)^{v_n}_{assign}, (\alpha_n)_{check}, \alpha_n'$ is plan for T'.*

We prove this lemma by showing that $s \otimes \alpha_1 \otimes \cdots \otimes \alpha_n$ is P-bisimilar to $s \otimes (\alpha_1)^{v_1}_{assign} \otimes (\alpha_1)_{check} \otimes \alpha_1' \otimes \cdots \otimes (\alpha_n)^{v_n}_{assign} \otimes (\alpha_n)_{check} \otimes \alpha_n'$. We prove this property by recurrence on n:

- $n = 0$: s is P-bisimilar to s.
- $n > 0$: we suppose that $s \otimes (\alpha_1)^{v_1}_{assign} \otimes (\alpha_1)_{check} \otimes \alpha_1' \otimes \cdots \otimes (\alpha_{n-1})^{v_{n-1}}_{assign} \otimes (\alpha_{n-1})_{check} \otimes \alpha_{n-1}' = s'_{n-1}$ is P-bisimilar to $s \otimes \alpha_1 \otimes \cdots \otimes \alpha_{n-1} = s_{n-1}$. First note that s_{n-1} is P-bisimilar to $s'_{n-1} \otimes (\alpha_n)^{v_n}_{assign}$ without $flagAssigned$ worlds, since the action $(\alpha_n)^{v_n}_{assign}$ without the event $e_{tagAssign}$ does not change any proposition on P, and epistemic relations are also preserved. Furthermore, the action $(\alpha_n)_{check}$ removes $flagAssigned$-worlds, thus s_{n-1} is P-bisimilar to $s'_{n-1} \otimes (\alpha_n)^{v_n}_{assign} \otimes (\alpha_n)_{check}$. The event $e_{tagLost}$ is then not applicable α_n', then it is equivalent to α_n with each formula φ replaced by p_φ. Since $(\alpha_n)_{check}$ ensured that p_φ is equivalent to φ in s'_{n-1}, we conclude that $s_{n-1} \otimes \alpha_n$ is P-bisimilar to $s'_{n-1} \otimes (\alpha_n)^{v_n}_{assign} \otimes (\alpha_n)_{check} \otimes \alpha_n'$.

To make the reduction polynomial, we apply the same idea as for the proof of Theorem 3.

Declaration of interests

[X] The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

□The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Thomas Bolander, 3 June 2020

AUTHOR DECLARATION TEMPLATE

We wish to draw the attention of the Editor to the following facts which may be considered as potential conflicts of interest and to significant financial contributions to this work. [OR] We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us.

We confirm that we have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property. In so doing we confirm that we have followed the regulations of our institutions concerning intellectual property.

We further confirm that any aspect of the work covered in this manuscript that has involved either experimental animals or human patients has been conducted with the ethical approval of all relevant bodies and that such approvals are acknowledged within the manuscript. [CAN BE DELETED IF NOT RELEVANT]

We understand that the Corresponding Author is the sole contact for the Editorial process (including Editorial Manager and direct communications with the office). He/she is responsible for communicating with the other authors about progress, submissions of revisions and final approval of proofs. We confirm that we have provided a current, correct email address which is accessible by the Corresponding Author and which has been configured to accept email fromtobo@dtu.dk.

Signed by all authors as follows:

Thomas Bolander

Sophie Pinchinat

François Schwarzentruber

AUTHOR DECLARATION TEMPLATE

We wish to draw the attention of the Editor to the following facts which may be considered as potential conflicts of interest and to significant financial contributions to this work. [OR] We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us.

We confirm that we have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property. In so doing we confirm that we have followed the regulations of our institutions concerning intellectual property.

We further confirm that any aspect of the work covered in this manuscript that has involved either experimental animals or human patients has been conducted with the ethical approval of all relevant bodies and that such approvals are acknowledged within the manuscript. [CAN BE DELETED IF NOT RELEVANT]

We understand that the Corresponding Author is the sole contact for the Editorial process (including Editorial Manager and direct communications with the office). He/she is responsible for communicating with the other authors about progress, submissions of revisions and final approval of proofs. We confirm that we have provided a current, correct email address which is accessible by the Corresponding Author and which has been configured to accept email fromtobo@dtu.dk.

Signed by all authors as follows:

Thomas Bolander

Sophie Pinchinat

François Schwarzentruber