On the Impact of Modal Depth in Epistemic Planning

Tristan Charrier  
Université de Rennes 1  
IRISA  
th Tristan.charrier@irisa.fr

Bastien Maubert  
Università degli studi di Napoli  
Federico II  
bastien.maubert@gmail.com

François Schwarzentruber  
ENS Rennes / IRISA  
francois.schwarzentruber @ens-rennes.fr

Abstract

Epistemic planning is a variant of automated planning in the framework of dynamic epistemic logic. In recent works, the epistemic planning problem has been proved to be undecidable when preconditions of events can be epistemic formulas of arbitrary complexity, and in particular arbitrary modal depth. It is known however that when preconditions are propositional (and there are no postconditions), the problem is between PSPACE and EXPTIME. In this work we bring two new pieces to the picture. First, we prove that the epistemic planning problem with propositional preconditions and without postconditions is in PSPACE, and is thus PSPACE-complete. Second, we prove that very simple epistemic preconditions are enough to make the epistemic planning problem undecidable: preconditions of modal depth at most two suffice.

Table 1: Overview (d: modal depth; gray: this paper).

<table>
<thead>
<tr>
<th>d = 0</th>
<th>no postconditions</th>
<th>PSPACE-complete</th>
<th>Decidable</th>
</tr>
</thead>
<tbody>
<tr>
<td>d ≤ 1</td>
<td>?</td>
<td>Undecidable</td>
<td></td>
</tr>
<tr>
<td>d ≤ 2</td>
<td>Undecidable</td>
<td>Undecidable</td>
<td></td>
</tr>
<tr>
<td>unbounded</td>
<td>Undecidable</td>
<td>Undecidable</td>
<td></td>
</tr>
</tbody>
</table>

On the one hand, when only propositional preconditions are used, such as $\varphi = \text{“Bob is married”}$, the problem is decidable if postconditions are also propositional [Yu et al., 2013]. In this case it is in $k$-EXPTIME, where $k$ is the maximal modal depth of goal formulas [Aucher et al., 2014]; if there are no postconditions (events are purely epistemic), the problem is in $\text{EXPSpace}$ [Bolander et al., 2015].

On the other hand, epistemic preconditions such as $\varphi = \text{“Bob considers it possible that Anne knows that Bob does not know that it is raining”}$ yield undecidability: if propositional postconditions are allowed, then the problem is already undecidable with preconditions of modal depth one [Bolander and Andersen, 2011]. It is also known to be undecidable without postconditions, if we allow for preconditions of unbounded modal depth [Aucher and Bolander, 2013].

See Table 1 for a summary of results about epistemic planning.

In this paper, our contribution is twofold:

1. With propositional preconditions and no postconditions, epistemic planning is in PSPACE (Theorem 1). The key point is that in this case events commute [Löwe et al., 2011]. This allows for a succinct representation of tuples of events, and we build upon a model checking procedure from [Aucher and Schwarzentruber, 2013] to devise a polynomial space decision procedure.

2. Epistemic planning without postconditions is already undecidable with preconditions of modal depth two (Theorem 2). The proof, by reduction from the halting problem for two counter machines, refines the one given in [Aucher and Bolander, 2013], which requires preconditions with unbounded modal depth. By designing more involved gadgets to code the configurations and instructions of the machines, we manage to bound the modal depth of preconditions.

1 Introduction

A key objective in artificial intelligence is to develop autonomous agents able to plan their actions towards achieving their goals, and to reason about their own and other agents’ knowledge. Planning, which consists in finding a sequence of actions to reach a given objective from an initial situation, is a central research domain in artificial intelligence. Concerning reasoning about knowledge, dynamic epistemic logic (DEL) is now recognised as a very promising framework [van Ditmarsch et al., 2007]. Recently, planning and dynamic epistemic logic have been combined in the so-called epistemic planning problem [Bolander and Andersen, 2011].

In DEL, events can deal with high-order reasoning. For instance we may model the following event:

“Anne receives a letter revealing that $\varphi$ is true and Bob knows that Anne receives the truth value of $\varphi$ but Anne is unsure whether Bob knows that or not.”

In DEL, $\varphi$ is called a precondition: $\varphi$ needs to be true for this event to occur, and therefore its occurrence brings the information that $\varphi$ is true. This event is purely informative, but DEL also allows physical (ontic) effects on the world; these are referred to as postconditions. One natural question is: how does the nesting of knowledge in pre- and postconditions impact the complexity of the epistemic planning problem?

This paper presents an overview of results about epistemic planning.
We first recall the background on epistemic planning in Section 2. We establish our two contributions, described above, in Section 3 and Section 4 respectively. We briefly discuss future work in Section 5.

2 Background on epistemic planning

In this section, we recall the necessary background about dynamic epistemic logic and epistemic planning.

2.1 Dynamic epistemic logic

Let $AP$ be a countably infinite set of atomic propositions, and let $Ag = \{1, \ldots, n\}$ be a finite set of agents. The epistemic language $L_{EL}$ is the language of propositional logic extended with one knowledge modality for each agent. Intuitively, $K_a \varphi$ reads as “agent $a$ knows that $\varphi$ holds”. The syntax of $L_{EL}$ is given by the following grammar:

$$
\varphi :: p | \neg \varphi | (\varphi \lor \varphi) | K_a \varphi, \text{ where } p \in AP \text{ and } a \in Ag.
$$

The semantics of $L_{EL}$ is given in terms of epistemic models that represent how the agents perceive the world.

**Definition 1** An epistemic model $M = (W, \{R_a\}_{a \in Ag}, V)$ is a tuple where:

- $W$ is a non-empty finite set of possible worlds,
- $R_a \subseteq W \times W$ is an accessibility relation for agent $a$,
- $V : AP \rightarrow 2^W$ is a valuation function.

We write $w \in M$ for $w \in W$, $|M|$ for $|W|$, and $(M, w)$ is called a pointed epistemic model. The intended meaning of $wR_aw'$ is that in world $w$ agent $a$ considers that $w'$ might be the actual world.

The semantics of $L_{EL}$ is defined as follows:

- $M, w \models p$ if $w \in V(p)$;
- $M, w \models \neg \varphi$ if it is not the case that $M, w \models \varphi$;
- $M, w \models (\varphi \lor \psi)$ if $M, w \models \varphi$ or $M, w \models \psi$;
- $M, w \models K_a \varphi$ if for all $w'$ s.t. $wR_aw'$, $M, w' \models \varphi$.

Dynamic epistemic logic (DEL) extends epistemic logic with modalities that represent occurrences of events. In DEL events are represented by event models, defined below. In general DEL events can bring information and modify the world, and such events are called ontic events [van Ditmarsch and Kooi, 2006]; in this work however we focus on purely informative events, called epistemic events [Baltag et al., 1998].

**Definition 2** An event model $E = (E, \{\rightarrow_a\}_{a \in Ag}, pre)$ is a tuple where:

- $E$ is a non-empty finite set of possible events,
- $\rightarrow_a \subseteq E \times E$ is an accessibility relation on $E$ for agent $a$,
- $pre : E \rightarrow L_{EL}$ is a precondition function.

We write $e \in E$ for $e \in E$, $|E|$ for $|E|$, and $(E, e)$ is called a pointed event model, where $e$ represents the actual event of $(E, e)$. An event $e$ can occur in a world $w$ of an epistemic model $M$ if, and only if, its precondition is verified, i.e. $M, w \models pre(e)$, which leads to the following definition:

**Definition 3** Given $M = (W, \{R_a\}_{a \in Ag}, V)$ an epistemic model and $E = (E, \{\rightarrow_a\}_{a \in Ag}, pre)$ an event model, the update product of $M$ and $E$ is the epistemic model $M \otimes E = (W^\otimes, \{R_a^\otimes\}_{a \in Ag}, V^\otimes)$ where:

$$
W^\otimes = \{(w, e) \in W \times E | M, w \models pre(e)\},
$$

$$
R^\otimes_a(w, e) = \{(w', e') \in W^\otimes | wR_aw' \text{ and } e \rightarrow_a e'\},
$$

$$
V^\otimes(p) = \{(w, e) \in W^\otimes | M, w \models p\}.
$$

The product of a pointed epistemic model $(M, w)$ with a pointed event model $(E, e)$ is defined as $(M, w) \otimes (E, e) := (M \otimes (E, w), e)$ if $M, w \models pre(e)$, otherwise it is undefined.

We can now define the syntax and semantics of DEL. The syntax is given by the following grammar:

$$
\varphi :: p | \neg \varphi | (\varphi \lor \varphi) | K_a \varphi | (\varphi \land \psi) | (\varphi \leftrightarrow \psi) | (\varphi \rightarrow \psi), \text{ where } p \in AP, a \in Ag \text{ and } (E, e) \text{ is a pointed event model.}
$$

The semantics is the same as for $L_{EL}$, with the following additional case:

- $M, w \models (E, e) \varphi$ if $M, w \models pre(e)$, and $(M, w) \otimes (E, e) \models \varphi$.

**Example 1** Consider the pointed epistemic model $(M, w)$ in Figure 1(a). Proposition $p$ is true in the actual world $w$ but both agents $a, b$ do not know that $p$ holds: $M, w \models \neg K_1p \land \neg K_2p$.

Figure 1(b) shows a pointed event model $(E, e)$ where the precondition of the actual event $e$ is $p$, and the one of event $f$ is $\top$. $(E, e)$ represents the event where agent 1 learns that $p$ is true while agent 2 believes that nothing happens. Figure 1(c) shows the product $(M, w) \otimes (E, e)$, which represents the situation after event $(E, e)$. Observe that agent 1 knows $p$ and agent 2 does not.

**Remark 1** We do not make any assumption on the nature of the accessibility relations in epistemic and event models.

2.2 Epistemic planning

Let $C$ be a class of pointed event models. The epistemic planning problem restricted to $C$ is the following:

**Definition 4** (Epistemic planning problem)

**Input**: a pointed epistemic model $(M, w)$, a finite set of pointed event models $E \subseteq C$, and an epistemic goal formula $\varphi_g$;

**Output**: yes if there exists a sequence of pointed event models $(E_1, e_1), \ldots, (E_p, e_p) \in E$ (a plan) such that $M, w \models (E_1, e_1) \ldots (E_p, e_p) \varphi_g$, no otherwise.

We now establish the precise complexity of this problem for propositional event models.
3 Propositional preconditions

Let $C_0$ be the class of (pointed) epistemic event models where
preconditions are propositional formulas. For instance, the
pointed event model depicted in Figure 1(c) is in $C_0$ since both
$p$ and $T$ are propositional formulas. The epistemic planning
restricted to $C_0$ is known to be PSPACE-hard [Bolander et al.,
2015]. We establish that it is actually PSPACE-complete.

As pointed out in [Löwe et al., 2011], epistemic event mod-
nels with propositional preconditions commute. Formally:

**Lemma 1** For all pointed epistemic models $(M, w)$, for
all pointed event models $(E_1, e_1)$ and $(E_2, e_2)$ in $C_0$,
$M \otimes E_1 \otimes E_2, (w, e_1, e_2)$ exists iff $M \otimes E_2 \otimes E_1, (w, e_2, e_1)$
exists, and in that case they are bisimilar.

As a consequence, in the rest of the section, the order in
which events are applied in an initial world is indifferent.
Only the number of times each event occurs is relevant, and
the proof of our result heavily relies on this property.

We first establish a preliminary result on the model check-
ning problem for a dedicated language: we extend the
dynamic epistemic language with iterations of event models
in $C_0$, that is, constructs of the form $(\langle E, e \rangle^i)\psi$ where $(E, e)$
is a pointed event model in $C_0$ and $i$ is a positive integer.
We suppose here that $i$ is written in binary so that this
language, called $L_{C_0}^i$, is exponentially more succinct than DEL.
Classically, the model checking problem for $L_{C_0}^i$ is, given
a pointed epistemic model $(M, w)$ and a formula $\Phi \in L_{C_0}^i$, to
decide whether $M, w \models \Phi$.

**Proposition 1** Model checking $L_{C_0}^i$ is in PSPACE.

**Proof** We design a deterministic algorithm that takes as
an input a pointed epistemic model $(M, w_0)$ and a formula
$\Phi \in L_{C_0}^i$, and decides whether $M, w_0 \models \Phi$. Without loss of
generality, we suppose that all event models appearing in
the formula are the same, noted $E = (E, e, pre)$ (if not, we
replace each one by their disjoint union). Let $e_1, \ldots, e_{|E|}$ be
an enumeration of the possible events in $E$. By Lemma 1, all
permutations of events in a tuple $(w, e_1, \ldots, e_{|E|})$ are equiva-
 lent in the sense that either they all are worlds in $M \otimes E^P$
and they all are bisimilar, or none of them exists: only the number
of times each event occurs is relevant. For a world $w$ and a
vector $\vec{n} = (n_1, \ldots, n_{|E|})$, we thus let $w \vec{n}$ denote the repre-
sentative permutation $(w, e_1 \times \overset{n_1}{\ldots} e_1, \ldots, e_{|E|} \times \overset{n_{|E|}}{\ldots} e_{|E|})$.

Let $mc$ be the algorithm given in Figure 2, and let $0^{\vec{n}}$ de-
note the null $|E|$-vector. We claim that $mc(M, w_0, 0^{\vec{n}}, \Phi)$
returns true iff $M, w_0 \models \Phi$. To prove this claim we establish
that for all $w \in M$, all integers $n_1, \ldots, n_{|E|}$ and all subfor-
ma $\varphi$ of $\Phi$, the following property $P$ holds:

If $M \otimes E^{\sum_{i=1}^{|E|} n_i}, w \vec{n}$ exists then
$mc(M, w, E, (n_1, \ldots, n_{|E|}), \varphi)$ returns true iff
$M \otimes E^{\sum_{i=1}^{|E|} n_i}, w \vec{n} \models \varphi$.

Property $P$ is proven by induction on $\varphi$. We omit the boolean
cases and case $(\langle E, e \rangle^i)\psi$ which are trivial.

Case $K_a \psi$: the algorithm has to check that $\psi$ holds in all
$a$-successors of $w \vec{n}$ in $M \otimes E^{\sum_{i=1}^{|E|} n_i}$. Every $a$-successor
of $w \vec{n}$ is a permutation of some $u \vec{e}$ and is bisimilar to it.

We thus need to enumerate all worlds $u$ and vectors $\vec{e}$ that
represent some $a$-successor, and verify that $\psi$ holds in $u \vec{e}$.

Given a tuple $u \vec{e}$, to check whether it is a permutation of some
$a$-successor of $w \vec{n}$, we first check that it is an exist-
ning world in $M \otimes E^{\sum_{i=1}^{|E|} n_i}$. Since events are purely
epistemic and propositional, preconditions of successive events
can all be checked in the initial world $w$. This is done by
calling function $preok(M, u, E, \vec{e})$, which checks that for all
$i \in \{1, \ldots, |E|\}$, if $\xi_i > 0$ then $pre(e_i)$ is true in $u$.

Next, we check that some permutation of $u \vec{e}$ is indeed a-
related to $w \vec{n}$: we should first have $u \in R_u(w)$; then, it
should be possible to map each occurrence of an event $e_i$
in $u \vec{n}$ to some occurrence of some $a$-related event $e_j$ in
$u \vec{e}$ so as to form a bijection. Deciding whether such a bij-
ection exists amounts to solving the following integer linear
program: checking whether there exist positive integers
$(x_{i,j})_{(i,j) \in \{(1,\ldots,|E|)\}^2 \in R_u(w)}$, where $x_{i,j}$ is the number of
times $e_j$ is chosen as $a$-successor for $e_i$, such that:

$$
\begin{cases}
\sum_{j \in U_{e_i}} x_{i,j} = n_i & \text{for all } i \in \{1, \ldots, |E|\}, \\
\sum_{i \in U_{e_j}} x_{i,j} = \xi_j & \text{for all } j \in \{1, \ldots, |E|\}.
\end{cases}
$$

This is done by calling $succe(E, a, \vec{n}, \vec{e})$.

**Spatial complexity.** The maximal number of nested calls
is bounded by $|\Phi|$, so that the number of local variables to be
stored is polynomial in $|\Phi|$. Next, the space used to store vec-
tor $\vec{n}$ in each call is in $O(|\Phi|^2)$ (see [Charrier et al., 2016] for
details). Finally, checking consistency of a system $S$ can
be done in non-deterministic time polynomial in the number
of bits needed to encode $\vec{n}$ and $\vec{e}$ [Papadimitriou, 1981], and
therefore in deterministic space polynomial in $|\Phi|$.

**Theorem 1** The epistemic planning problem restricted to $C_0$
is in PSPACE.

**Proof** We adapt the algorithm given in [Bolander et al.,
2015] (Theorem 5.8). First it is proved in [Sadzik, 2006] that,
noting $\approx_d$ the $d$-bisimulation\footnote{Bisimulation up to modal depth $d$.} for
event models (see [Bolander et al., 2015; Sadzik, 2006; van Ditmarsch et al.,
2007]), for every $d \geq 0$, every pointed event model $(E, e)$ is $\approx_d$
stabilizing at iteration $|E|^d$; formally, $(E, e)^k \approx_d (E, e)^{k+1}$
for all $k \geq |E|^d$\footnote{Actually a better bound is proved in [Bolander et al., 2015].}. Secondly, by Lemma 1, event models with
propositional preconditions commute. Therefore, the follow-
ing algorithm correctly solves the epistemic planning prob-
lem for event models with propositional preconditions:

Given input $(\langle M, w \rangle, \{(E_1, e_1), \ldots, (E_m, e_m)\}, \varphi_g)$:

1. Compute $d$, the modal depth of the goal formula $\varphi_g$;
2. For each $i \in \{1, \ldots, m\}$, non-deterministically guess
   $n_i \in \{0, \ldots, |E_i|^d\}$;
3. Accept if $M, w \models (\langle E_1, e_1 \rangle^{n_1}) \times \cdots \times (\langle E_m, e_m \rangle^{n_m}) \varphi_g$.

This algorithm is non-deterministic. The first step is clearly
performed in space polynomial in the size of the input. Con-
cerning the second point, each $n_i$ can be exponential in $d$.
and thus in $|\varphi|$, but its binary representation uses polynomial space. Since $\langle (E_1, e_1)^{m_1} \ldots (E_m, e_m)^{m_m} \rangle \varphi_g$ is an $L^{\ell \ell}$ formula, it follows from Proposition 1 that the last step can also be performed in polynomial space. The epistemic planning problem restricted to $C_0$ is therefore in NPSPACE and thus in PSPACE by Savitch’s theorem [Savitch, 1970].

We now turn to the case of modal preconditions with bounded modal depth.

## 4 Preconditions of bounded modal depth

Let $C_2$ be the class of event models with preconditions of modal depth at most two. In this section, we prove the following theorem by reducing the problem of halting two-counter machines to $C_2$.

**Theorem 2** The epistemic planning problem restricted to $C_2$ is undecidable.

We first recall the halting problem for two-counter machines, known to be undecidable [Minsky, 1967], and then we reduce it to the epistemic planning problem restricted to $C_2$.

### 4.1 Two-counter machines

We present two-counter machines as introduced in [Minsky, 1967].

**Definition 5** A two-counter machine $M$ is a sequence of instructions $(I_0, \ldots, I_N)$ where

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

We call program line a pair $k : I_k$.

**Example 2** The four program lines shown on the right define a two-counter machine $M_{\text{ex}}$.

- 0:inc(1)
- 1:gotocond(1, 3)
- 2:goto(0)
- 3:halt

A configuration of a two-counter machine $M$ is a triple $(\ell, c_1, c_2)$ where $\ell \in \{0, \ldots, N\}$ is the program counter and $c_1, c_2 \in \mathbb{N}$ are the two data counters. Let $C_M = \{0, \ldots, N\} \times \mathbb{N} \times \mathbb{N}$ be the set of all possible configurations.

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

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

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 $\rightarrow_M$. For instance, the machine $M_{\text{ex}}$ given in Example 2 above does not halt. The halting problem for two-counter machines consists in deciding, given a two-counter machine, whether it halts or not. This problem is well known to be undecidable [Minsky, 1967].

### 4.2 The reduction

We define an effective reduction $tr$ that, given a two-counter machine $M$, computes an instance $tr(M)$ of the epistemic planning problem restricted to $C_2$. We fix $M$ and the rest of the section is devoted to defining $tr(M) = (\langle M_0, w_0 \rangle; E; \varphi_g)$ and justifying its correctness (Proposition 2). As in [Aucher and Bolander, 2013], we only use one agent $a$ (□ stands for $K_a$ and ◊ stands for $\neg K_a$), configurations of $M$ are represented by pointed epistemic models, and the initial pointed epistemic model represents the initial configuration $(0, 0, 0)$. Each program line $l : I_\ell$ is represented by one or two pointed event model(s), such that a plan corresponds to a sequence of program lines. The goal formula expresses that the final pointed epistemic model represents a halting configuration.

**Pointed epistemic models**

Let $(\ell, c_1, c_2)$ be a configuration of $M$. We describe the pointed epistemic model $(\langle M(\ell, c_1, c_2), w(\ell, c_1, c_2) \rangle)$ (shortened
Lemma 2. A pointed epistemic model \((\mathcal{M}, w)\) is valid if \(w\) has an \(a\)-child for each \(\ell \in \{0, \ldots, N\}\) and a \(p\)-child for each \(i \in \{1, 2\}\).

Note that by definition, every configuration model is valid. Further down, we will define event models of \(E\) such that:

**Lemma 2** For every configuration \((\ell, c_1, c_2)\), it holds that \((\mathcal{M}, w)_{(\ell, c_1, c_2)}\) is isomorphic\(^3\) to \((\mathcal{M}, w)_{(\ell', c_1', c_2')}\) if \((\ell, c_1, c_2) \rightarrow_M (\ell', c_1', c_2')\).

\[^3\]More precisely, the reachable parts of the pointed epistemic models are isomorphic.
in Figure 7) is meant to increment the data counter $c_1$. The intermediate event of precondition $p_1 \land \Diamond q_1$ duplicates once the $p_1$-child of the root: it adds one $p_1$-world at the start of the $p_1$-chain. The last group leaves data counter $c_2$ unchanged.

- The product $(M, w)_{(t, c_1, c_2)} \otimes (E, e)_{\ell: inc(1)}$ is isomorphic to $(M, w)_{((t + \text{lenen}, c_1, c_2)}$.
- For the same reason as for $(E, e)_{\ell: gotocond(1)}$, the product $(M, w)_{(t', c_1, c_2)} \otimes (E, e)_{\ell: inc(1)}$ with $t' \neq \ell$ is not valid.

**Event models for $\ell: gotocond(i, \ell')$.** Figure 9 describes models $(E, e)_{\ell: gotocond(1, \ell')}$ and $(E_{\ell: inc(1)}, e_{\ell: inc(1)})_{\ell: gotocond(1, \ell')}$. They mimic the effect of $\ell: gotocond(1, \ell')$ in case $c_1 = 0$ and case $c_1 > 0$, respectively (for $\ell: gotocond(2, \ell')$, constructions are symmetric).

- $(M, w)_{(t, 0, c_2)} \otimes (E, e)_{\ell: gotocond(1, \ell')}$ is isomorphic to $(M, w)_{(t', c_1, c_2)}$: indeed, precondition $\neg (p_1 \land \neg q_1)$ checks that the $p_1$-chain in the data counter $c_1$ group is of length 0. Here it is the case, so the data counter group $c_1$ remains unchanged. However, when $c_1 > 0$, the $p_1$-child of the root of $(M, w)_{(t, c_1, c_2)}$ violates this precondition. It is thus removed, so that the product $(M, w)_{(t, c_1, c_2)} \otimes (E, e)_{\ell: gotocond(1, \ell')}$ is not valid.

- $(M, w)_{(t, 0, c_2)} \otimes (E_{\ell: inc(1)}, e_{\ell: inc(1)})_{\ell: gotocond(1, \ell')}$ with $c_1 > 0$ is isomorphic to $(M, w)_{(t + \text{lenen}, c_1, c_2)}$. Indeed, portion $\text{shorten}(1)$ (Figure 7) is meant to decrement data counter $c_1$ by one: precondition $p_1 \land \neg q_1 \land \lozenge T$ checks that we are in the $p_1$-chain ($p_1$), but not at the start ($\neg q_1$) nor the end ($\lozenge T$) of the chain. The last world of the $p_1$-chain is thus removed when $c_1 > 0$. When $c_1 = 0$, precondition $\lozenge (p_1 \land \neg q_1)$ is violated by the $p_1$-child of the root of $(M, w)_{(t, 0, c_2)}$: indeed, this precondition checks that the length of the $p_1$-chain is at least 1. The product $(M, w)_{(t, 0, c_2)} \otimes (E_{\ell: inc(1)}, e_{\ell: inc(1)})_{\ell: gotocond(1, \ell')}$ is thus not valid.

- For $\ell' \neq \ell$, $(M, w)_{(t', c_1, c_2)} \otimes (E, e)_{\ell: gotocond(1, \ell')}$ and $(M, w)_{(t', c_1, c_2)} \otimes (E_{\ell: inc(1)}, e_{\ell: inc(1)})_{\ell: gotocond(1, \ell')}$ are not valid.

Note that the product of a non-valid pointed epistemic model with any pointed event model is not valid since no event model can create the missing children of the root without using postconditions.

**Goal formula**

The goal formula $\varphi_g$ in $tr(M)$ is $\varphi_{\text{valid}} \land \varphi_{\text{halt}}$, where:

- $\varphi_{\text{valid}} := \bigwedge_{t=0}^{N} a_t \land \lozenge q_1 \land \neg q_2$, and
- $\varphi_{\text{halt}} := \lozenge (a_N \land \square \bot)$.

**Proposition 2** $M$ halts if there is a plan for $tr(M)$.

The proof can be found in [Charrier et al., 2016].

### 4.3 Comparison

In [Aucher and Bolander, 2013] the program counter as well as the data counters are represented with chains of worlds, and incrementation, decrementation and replacement of a value by another one are implemented on such chains. While the first two operations can be performed with preconditions of modal depth two, $\text{repl}(\ell, \ell')$ requires unbounded nesting in general to be implemented on chains. We observed that unlike data counters, the program counter is bounded so that we can avoid chains for its representation, and provide an alternative gadget for $\text{repl}(\ell, \ell')$ that only uses preconditions of modal depth two.

### 5 Future work

The natural continuation is to complete Table 1. First, is the epistemic planning problem decidable for preconditions of modal depth one and no postconditions, or do modalities in preconditions immediately bring about undecidability? Second, what is the exact complexity of the problem with propositional pre- and postconditions? It is known to be decidable [Yu et al., 2013], with a non-elementary upper bound [Aucher et al., 2014] and a PSPACE lower bound [Bolander et al., 2015]; this a big gap that should be bridged.

### References
