Planning based on dynamic epistemic logic

Benedikt Löwe[1,2,3], Eric Pacuit[4], Andreas Witzel[5]

[1] Institute for Logic, Language and Computation, Universiteit van Amsterdam, Postbus 94242, 1090 GE Amsterdam, The Netherlands; b.loewe@uva.nl
[2] Department Mathematik, Universität Hamburg, Bundesstrasse 55, 20146 Hamburg, Germany
[3] Mathematisches Institut, Rheinische Friedrich-Wilhelms-Universität Bonn, Endenicher Allee 60, 53115 Bonn, Germany
[4] Tilburg Center for Logic and Philosophy of Science, Tilburg Universiteit van Tilburg, Postbus 90153, 5000 LE Tilburg, The Netherlands; E.J.Pacuit@uvt.nl
[5] Bioinformatics Group, Courant Institute of Mathematical Sciences, New York University, 715 Broadway, New York, NY 10003, United States of America; awitzel@nyu.edu

1 Introduction

In recent years, logicians have become increasingly interested in the interplay of several agents, especially in dynamic settings of epistemic and doxastic nature. Fruitful technical advances have led to deep insights into the epistemic nature of interaction by logicians, and this new direction of logic has quickly gained momentum and support, allowing a re-assessment of the role of logic for artificial intelligence. The complete formalization of concrete games in which knowledge and belief play a crucial role (called "knowledge games" by van Ditmarsch [24]) in terms of dynamic epistemic logic has been hailed as a great success. Van Eijck's software DEMO [26] renders this theoretical success into a useful tool. The first and second author have proposed to use formalisms based on epistemic logic for the formalization of narratives [11, 12], and the third author (in collaboration with Kennerly and Zvesper; [28, 10]) has formulated a simple knowledge-based action situation from a computer game. We formalize this (toy) example using dynamic epistemic logic (DEL) and explain the underlying planning problem, motivating the definition of a general *DEL planning problem* in which we propose to use DEL as a general formalism to generate intelligent and convincing social behavior in simulated characters.

Related Work.

The potential to use dynamic epistemic logic for actual implementations of reasoning processes permeates the literature; many papers mention concrete applications as the motivation for studying dynamic epistemic logic. The DEMO software of Jan van Eijck [26] makes this very concrete; it has been used, e.g., for the Russian Card problem in [17, chapter 6].

Renardel de Lavalette and van Ditmarsch [14] discuss updating and maintaining a minimal epistemic model and identifying subclasses of DEL for which

that is possible. They provide model minimization for so-called simple actions in order to allow efficient model checking. Our scenario used in this paper has non-simple (though propositional) actions, and uses non-S5 models.

The closest cognate to our proposal is the work by van der Hoek and Wooldridge [23] on planning with epistemic goals, based on the idea of Giunchilia and Traverso to use model checking as a planning algorithm [9]. Their planning algorithm is based on S5 ATEL (for more on embedding DEL into temporal logics, cf. § 2), using a similar model as we did in [10]; we argue here that DEL provides a more flexible framework.

Related to the idea of using DEL for planning is the paper [1] on public announcement logic which examines which public announcement to make in a strategic setting with goals (assuming truthfulness).

Outline of this paper

In § 2 we shall give a standard introduction to dynamic epistemic logic, following roughly the textbook [25]. At the end of the section, we stress that DEL is not a temporal logic and comment on possible embeddings of DEL in temporal settings. In § 3, we return to the setting of the doxastically enriched computer game *Thief* from [28, 10] and give a formalization in terms of DEL which is then used in §§ 3.4 and 3.5 to explain possible applications (resticted to the case of our toy example). Based on this concrete example, we then define the *DEL planning problem* in § 4, and close the paper with a few pointers to future work in § 5.

2 Dynamic Epistemic Logic

Epistemic logic proved to be a powerful tool for studying distributed algorithms and multi-agent systems (cf. [7, 13]). Much of the current research on epistemic logic focuses not only on developing logics for reasoning about the knowledge and beliefs of interacting rational agents, but also the *dynamics* of information and beliefs (cf. [19] for an account of this perspective). In this section, we give an overview of product updates due to Baltag, Moss and Solecki [3] for the non-expert reader, following closely the textbook [25] (where the reader can find more details).

Let $\mathcal{A}$ be a finite set of agents and At a set of atomic propositions. An **epistemic model** is a tuple $\langle W, \{R_i\}_{i\in\mathcal{A}}, V\rangle$ where $W =: \mathrm{D}(\mathcal{M})$ is a non-empty set called the **domain of** $\mathcal{M}$, for each $i \in \mathcal{A}$, $R_i \subseteq W \times W$ is a binary relation on W (typically an equivalence relation) and $V : \mathsf{At} \to 2^W$ is a valuation function. If $w \in \mathrm{D}(\mathcal{M})$, we call $(\mathcal{M}, w)$ a **pointed epistemic model**. If w is clear from the context, we may omit it from the notation. The elements of W constitute "states of the world" and the relations R_i are **accessibility relations**, i.e., for states $w, v \in W$, wR_iv means "in state w, agent i would consider state v possible."

For example, consider two players i and j standing in front of a locked door. There are two codes c_1 and c_2 that potentially open the door. We model this by a state C_1 in which c_1 is the correct code and a state C_2 in which c_2 is the correct

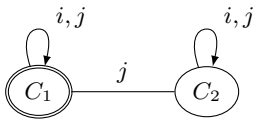


Fig. 1: The situation of two possible codes to open the door. The actual state is marked by double borders. Bidirectional arrows are displayed as plain line.

code. Suppose that c_1 is actually the correct code, agent i has this information but agent j does not have this information. This is depicted in Figure 1. Notice that the figure represents not only the information the players have about the value of the secret code, but also the information the *other* player has about the whether the other player is informed about the secret code. This can be made precise using the following logic.

The set of multiagent epistemic formulas, denoted $\mathcal{L}_{\mathcal{A}}$, is the smallest set of formulas generated by the following grammar:

$$P \mid \neg\varphi \mid \varphi \wedge \psi \mid \Box_i\varphi$$

where $P \in \mathsf{At}$ and $i \in \mathcal{A}$. We use the usual abbreviations for the other propositional connectives $(\vee, \rightarrow, \leftrightarrow)$, and we use $\mathcal{L}_{\mathrm{Prop}}$ to denote the propositional sub-language (i.e., not containing $\Box_i$). Truth of formulas $\varphi \in \mathcal{L}_{\mathcal{A}}$ is defined as usual in Kripke models.[6]

Returning to our example (and using the notation C_1 and C_2 to denote the propositions "c_1 is the correct code" and "c_2 is the correct code" as well as the corresponding states), the reader can verify that the following formulas are true at state C_1:

1. $\Box_i C_1$: *i knows the secret code is c_1*
2. $\neg\Box_j C_1$: *j does not know the secret code is c_1*
3. $\Box_j(\Box_i C_1 \vee \Box_i C_2)$: *$j$ knows that i knows the value of the secret code*
4. $\Box_i\neg\Box_j\Box_i C_1$: *$i$ knows that j does not know that i knows that the secret code is c_1.*

An **event model** $\mathcal{E}$ is a tuple $\langle S, \{\rightarrow_i\}_{i\in\mathcal{A}}, \mathsf{pre}\rangle$, where S is a nonempty set, for each $i \in \mathcal{A}$, $\rightarrow_i \subseteq S \times S$ is i's **accessibility relation**, and $\mathsf{pre} : S \rightarrow \mathcal{L}_{\mathcal{A}}$ is the **pre-condition function**. The set S is called the domain of $\mathcal{E}$, denoted $\mathrm{D}(\mathcal{E})$. We call $\mathcal{E}$ **propositional** if pre goes into $\mathcal{L}_{\mathrm{Prop}}$, i.e., all preconditions are propositional. The **product update** operation updates an epistemic model $\mathcal{M} = \langle W, \{R_i\}_{i\in\mathcal{A}}, V\rangle$ with an event model $\mathcal{E} = \langle S, \{\rightarrow_i\}_{i\in\mathcal{A}}, \mathsf{pre}\rangle$ and is defined as $\mathcal{M} \otimes \mathcal{E} = \langle W', \{R'_i\}_{i\in\mathcal{A}}, V'\rangle$ with

(i) $W' = \{(w, e) \mid w \in W, e \in S \text{ and } \mathcal{M}, w \models \mathsf{pre}(e)\}$,
(ii) $(w, e)R_i(w', e')$ iff wR_iw' in $\mathcal{M}$ and $e \rightarrow_i e'$ in $\mathcal{E}$, and
(iii) $V'((s, e)) = V(s)$.

[6] I.e., $\mathcal{M}, w \models \Box_i\varphi$ iff for all $v \in W$, if wR_iv then $\mathcal{M}, v \models \varphi$.

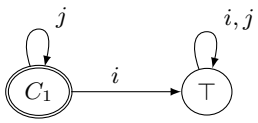


Fig. 2: The event model of agent j secretly learning that the correct code is c_1. We shall denote the left event by e_1 and the right event by e_2.

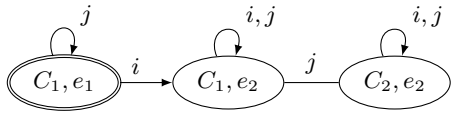


Fig. 3: The result of the epistemic update of the epistemic model from Figure 1 by the event model from Figure 2.

For pointed models, the point of the product is the pair of the factors' points.

If $\mathcal{E}$ and $\mathcal{E}'$ are two event models, we can analogously define the **product event model** $\mathcal{E} \otimes \mathcal{E}'$. It satisfies a type of associative law $(\mathcal{M} \otimes \mathcal{E}) \otimes \mathcal{E}' \equiv \mathcal{M} \otimes (\mathcal{E} \otimes \mathcal{E}')$, where $\equiv$ denotes isomorphism.

The usual notion of equivalence used in modal logic is the weaker notion of **bisimulation** [25, Definition 2.14], denoted $\leftrightarroweq$. Bisimilar pointed models are equivalent in the sense that they satisfy exactly the same formulas. For any model $\mathcal{M}$ and events $\mathcal{E}$, $\mathcal{E}'$, if $\mathcal{E} \leftrightarroweq \mathcal{E}'$ then $\mathcal{M} \otimes \mathcal{E} \leftrightarroweq \mathcal{M} \otimes \mathcal{E}'$ (note that the converse does not hold, see [27]). For each model $\mathcal{M}$, the union of all bisimulations of $\mathcal{M}$ with itself is again a bisimulation. The quotient structure of this bisimulation gives us the most compact model satisfying the same formulas, called the **bisimulation contraction**. By $\lfloor \mathcal{M} \rfloor$, we denote the cardinality of the domain of the bisimulation contraction of $\mathcal{M}$.

Returning to our example, Figure 2 shows the event where j *secretly* learns the code is c_1. It has two primitive events e_1 and e_2. The precondition of e_1 is C_1 (denoted $\mathsf{pre}(e_1) = C_1$) while the precondition for e_2 is $\top$ (denoted $\mathsf{pre}(e_2) = \top$). So e_2 reflects an event where "nothing is happening". Agent j is aware that event e_1 took place while agent i thinks that nothing happened (and is unaware that agent j learned the secret code).

Using the product update operation, we can construct the epistemic model shown in Figure 3, which reflects the situation after the event has occurred. As expected, we have that both i and j know the secret key (formally, $\Box_i C_1 \wedge \Box_j C_1$ is true at the state (C_1, e_1)). Furthermore, i does not know that j knows the value of the secret key (formally, $\neg\Box_i\Box_j C_1$ is true at (C_1, e_1)). In fact, i actually mistakenly believes that j does not know the value: $\Box_i\neg\Box_j C_1$.[7]

[7] Note that we are here using dynamic *epistemic* logic for a situation in which we describe *beliefs*. The main difference between knowledge and belief is that beliefs can be false whereas knowledge—in standard formalizations—cannot. If i discovers that j *does* know, the product update will produce a model in which i considers *no*

Now, adding to $\mathcal{L}_{\mathcal{A}}$ a modal operator $\langle \mathcal{E}, e \rangle$ for each pointed event model $\mathcal{E}, e$, we obtain the language $\mathcal{L}_{\mathrm{DEL}}$. Truth for these modalities is defined as

$$\mathcal{M}, w \models \langle \mathcal{E}, e \rangle \varphi \text{ iff } \mathcal{M}, w \models \mathsf{pre}(e) \text{ and } \mathcal{M} \otimes \mathcal{E}, (w, e) \models \varphi.$$

Given a pointed model $(\mathcal{M}, w)$ and a formula $\varphi \in \mathcal{L}_{\mathcal{A}}$, checking $\mathcal{M}, w \models \varphi$ can be done in time polynomial in the size of $\mathcal{M}$ and φ [22]. Some care must be taken with respect to the product update, as at first glance it can potentially lead to exponentially growing models (cf. § 3.3).

Given that DEL is about *dynamics*, i.e., the change of an epistemic situation through time, it may come as a slight surprise that DEL is not a temporal language, i.e., has no means of expressing temporal relations. In order to have temporal relations, one can embed DEL into an appropriate temporal logic. In [20], the authors give a natural embedding of DEL into ETL. Since the details of temporal logic do not matter for our purposes, we think of the temporal setting as follows: Fix a pointed epistemic model $(\mathcal{M}, w)$ and a finite set of (pointed) event models $\mathfrak{E}$. For prefixes of finite sequences $\sigma := (\mathcal{E}_0, \ldots, \mathcal{E}_n)$ of models in $\mathfrak{E}$, we have a natural notion of **immediate successor**, viz. extension by one additional model. Let $\mathcal{M} \otimes \sigma := \mathcal{M} \otimes \mathcal{E}_0 \otimes \ldots \otimes \mathcal{E}_n$, then the collection of these epistemic models forms a tree structure with successor structure derived from the finite sequences and $\mathcal{M} = \mathcal{M}_{\varnothing}$ at the root. It is this tree structure that we consider to be the natural temporal setting for DEL planning (cf. § 4).

Slightly more precisely, for a pointed epistemic model $(\mathcal{M}, w)$ and $(\mathcal{E}, e) \in \mathfrak{E}$, we say that $(\mathcal{E}, e)$ is **possible at** $(\mathcal{M}, w)$ if $\mathcal{M}, w \models \langle \mathcal{E}, e \rangle \top$. We say that a sequence σ is **legal** if it is empty or its (uniquely determined) immediate predecessor σ^* is legal and $(\mathcal{E}^*, e^*)$ is possible at $\mathcal{M} \otimes \sigma^*$, with $(\mathcal{E}^*, e^*)$ being the last element of σ. The set of legal sequences, denoted LS, contains exactly those sequences that can be performed in the given order, since the preconditions of each event are met at the appropriate time. They form a subtree of our tree structure. If we want to impose further external restrictions on the possible courses of action, we can consider a subtree $T \subseteq \mathrm{LS}$.

To conclude our description of DEL, we should note that we do not consider events that change actual facts (i.e., the valuation function). This is a serious restriction but doesn't affect the example in § 3. The definition of the product update can be extended to deal with factual change [22], but for the sake of simplicity, we restricted ourselves to purely epistemic events (cf. § 5).

3 An implementation project

We envision a use of DEL as a general engine (called "knowledge module" by Kennerly, Witzel, and Zvesper in [28, 10]) which allows for flexible specification of situations and events and then maintains the agents' mental models throughout

state of the world possible. For a more graceful handling and revising of inconsistent beliefs, we could use dynamic doxastic logic (DDL, cf. [4]). Since DEL works for the simple examples in this paper, we ignore these issues for the present discussion.

the progress of the game or scene, much like a physics engine maintains a model of the physical state of the world. We will illustrate this in a slightly richer scenario taken from an actual computer game. Note that, while we focus on maintaining one central epistemic model as part of the simulation engine, such a model can also be distributed and maintained by the individual agents [2].

We want to emphasize that this is merely a toy example, and while there is no claim to scalability we do want to point out some initial issues and possible approaches to handling them.

3.1 Thief: The Dark Project

The video game *Thief: The Dark Project*$^{\mathrm{TM}}$ by Eidos Interactive (1998) is themed as a game of stealth, in which the player (the thief) avoids being detected by computer-simulated guards. The player exploits the guard's—possibly mistaken—beliefs about the thief's presence. Due to the simplicity of the guard's control program, the guard's beliefs are in practice perceived as either "the thief is here" or "the thief is not here". The entertainment value of a typical *Thief* scenario could be enhanced by a guard that acts not only based on these two basic beliefs, but also depending on what he believes the thief believes, including what he believes the thief believes he believes.

3.2 Example scenario

We consider a minimalistic example scenario with a thief and a guard, modeling the agents' belief states as events occur in the game. In particular, we consider these ways in which relevant beliefs can come about: by causing or hearing noise, seeing the other one from behind, or facing each other.

We assume that the scene starts with thief and guard present, each uncertain of the other's presence, and that agents cannot enter or leave for the sake of simplicity. In our formalization, we consider the following kinds of events:

- $n_{\mathrm{t}}, n_{\mathrm{g}}$: The thief (the guard) makes some noise.
- $b_{\mathrm{t}}, b_{\mathrm{g}}$: The thief (the guard) sees the other one from behind.
- f: Thief and guard see each other face to face.

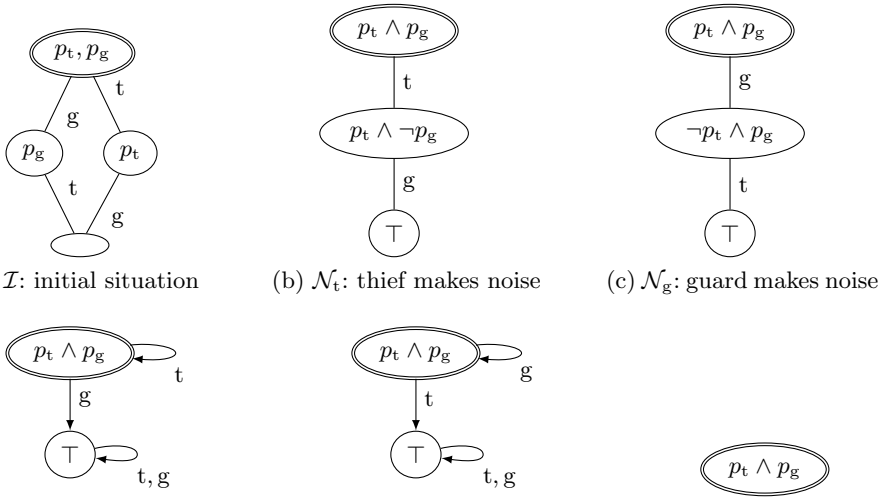
The intuitive epistemic effects of these events are as follows:

- n_{t}: The guard learns that a thief is present; the thief learns that, if a guard is present, that guard learns that the thief is present.
- b_{t}: The thief learns that a guard is present; the guard believes nothing has happened (he is not paranoid enough to constantly suspect being seen from behind)
- f: Thief and guard commonly learn that both are present.

The effects of n_{g} and b_{g} are analogous.

3.3 DEL formalization

To model this situation in DEL, we use the set of atomic propositions $\mathsf{At} = \{p_\mathrm{t}, p_\mathrm{g}\}$, with the reading that the thief, respectively the guard, is present. We formalize the initial situation by the pointed model $\mathcal{I}$ and the events described above by the set of pointed event models $E = \{\mathcal{N}_\mathrm{t}, \mathcal{N}_\mathrm{g}, \mathcal{B}_\mathrm{t}, \mathcal{B}_\mathrm{g}, \mathcal{F}\}$, as depicted in Figure 4. From now on we omit the qualifier "pointed".



(a) $\mathcal{I}$: initial situation (b) $\mathcal{N}_\mathrm{t}$: thief makes noise (c) $\mathcal{N}_\mathrm{g}$: guard makes noise

(d) $\mathcal{B}_\mathrm{t}$: thief sees guard from behind (e) $\mathcal{B}_\mathrm{g}$: guard sees thief from behind (f) $\mathcal{F}$: face to face

Fig. 4: Models for initial situation and events. Undirected edges represent bidirectional accessibilities. In models without directed edges, reflexive accessibilities are omitted.

Given that our aim is to maintain a model of the current situation as the game proceeds, we need to worry about the size of that model, since each product update could potentially multiply the number of states.

To analyze the situation, we first introduce some notions. We call an event model $\mathcal{E}$ **self-absorbing** if for all models $\mathcal{M}$ we have $\mathcal{M} \otimes \mathcal{E} \otimes \mathcal{E} \leftrightarroweq \mathcal{M} \otimes \mathcal{E}$, and we call two event models $\mathcal{E}, \mathcal{E}'$ **commutative** if for all models $\mathcal{M}$ we have $\mathcal{M} \otimes \mathcal{E} \otimes \mathcal{E}' \leftrightarroweq \mathcal{M} \otimes \mathcal{E}' \otimes \mathcal{E}$. The events we consider in the example will be both self-absorbing and commutative, enabling us to show that the model is well-behaved and stays small.

To show the results formally and for a more general class of events than the small set that we consider, we need one auxiliary notion. We call an event model $\mathcal{E}$ **almost-mutex** if there is at most one atomic event $e_\top \in \mathrm{D}(\mathcal{E})$ with $\mathsf{pre}(e_\top) = \top$ and $e_\top \rightarrow_i e_\top$ for all $i \in \mathcal{A}$, and the formulas $\mathsf{pre}(e)$ with $e \neq e_\top$ are pairwise inconsistent. We are now ready to state some results.

Lemma 1 *Propositional event models are commutative.*

Proof. Let $\mathcal{E}, \mathcal{E}'$ be propositional event models. We prove that $\mathcal{E} \otimes \mathcal{E}' \leftrightarrows \mathcal{E}' \otimes \mathcal{E}$. To see that this holds, consider the smallest relation ρ with $(s, s')\rho(s', s)$ for all $(s, s') \in \mathrm{D}(\mathcal{E}) \times \mathrm{D}(\mathcal{E}')$. This relation is a bisimulation due to commutativity of logical conjunction. □

Lemma 2 *Almost-mutex event models with transitive accessibility relations are self-absorbing.*

Proof. Let $\mathcal{E} = \langle S, \{\rightarrow_i\}_{i\in\mathcal{A}}, \mathsf{pre}\rangle$ be an almost-mutex event model with point $s \in S$. We again prove that $\mathcal{E} \otimes \mathcal{E} \leftrightarrows \mathcal{E}$. Consider the smallest relation $\rho \subseteq (\mathrm{D}(\mathcal{E}) \times \mathrm{D}(\mathcal{E})) \times \mathrm{D}(\mathcal{E})$ such that

$$(e, e)\rho e \qquad (e, e_\top)\rho e \qquad (e_\top, e)\rho e$$

for all $e \in \mathrm{D}(\mathcal{E})$. We show that this is a bisimulation on the submodels of $\mathcal{E} \otimes \mathcal{E}$ and $\mathcal{E}$ generated by (s, s) and s. To see this, note first that $(s, s)\rho s$.

Next, assume that $(e_1, e_2)\rho e$ and $(e_1, e_2) \rightarrow_i (e_1', e_2')$. We have to show that there is e' with $e \rightarrow_i e'$ and $(e_1', e_2')\rho e'$. By definition of ρ, we are in one of three cases:

- $e_1 = e_2 = e$. From $(e, e) \rightarrow_i (e_1', e_2')$ it follows that $e \rightarrow_i e_1'$ and $e \rightarrow_i e_2'$. If $e_1' = e_2'$ then $(e_1', e_2')\rho e_1'$ by definition of ρ and we are done. Otherwise $e_1' \neq e_2'$. Since $(e_1', e_2') \in \mathrm{D}(\mathcal{E}\otimes\mathcal{E})$, $\mathsf{pre}(e_1')$ and $\mathsf{pre}(e_2')$ cannot be inconsistent, and with $\mathcal{E}$ being almost-mutex it follows that one of the two events is $e_\top$. If $e_1' = e_\top$ then $(e_1', e_2')\rho e_2'$ by definition of ρ, and analogously if $e_2' = e_\top$.
- $e_1 = e$ and $e_2 = e_\top$. From $(e, e_\top) \rightarrow_i (e_1', e_2')$ it follows that $e \rightarrow_i e_1'$ and $e_\top \rightarrow_i e_2'$. If $e_1' = e_2'$ then $(e_1', e_2')\rho e_1'$ by definition of ρ. Otherwise $e_1' \neq e_2'$. Since $(e_1', e_2') \in \mathrm{D}(\mathcal{E} \otimes \mathcal{E})$, $\mathsf{pre}(e_1')$ and $\mathsf{pre}(e_2')$ cannot be inconsistent, and with $\mathcal{E}$ being almost-mutex it follows that one of the two events is $e_\top$. If $e_2' = e_\top$ then $(e_1', e_2')\rho e_1'$ by definition of ρ and we are done since $e \rightarrow_i e_1'$. Otherwise $e_1' = e_\top$, and from $e \rightarrow_i e_1' = e_\top \rightarrow_i e_2'$, by transitivity we get $e \rightarrow_i e_2'$. By definition of ρ, $(e_\top, e_2')\rho e_2'$.
- $e_1 = e_\top$ and $e_2 = e$. Analogous to the previous case.

Finally, assume that $(e_1, e_2)\rho e$ and $e \rightarrow_i e'$. We have to show that there is (e_1', e_2') with $(e_1, e_2) \rightarrow_i (e_1', e_2')$ and $(e_1', e_2')\rho e'$. This is easy to see with a similar case distinction as above, noting that $e_\top \rightarrow_i e_\top$ by assumption. □

For a sequence $\sigma = \mathcal{E}_1 \dots \mathcal{E}_k$ of event models, let $\mathrm{Set}(\sigma) = \{\mathcal{E}_1, \dots, \mathcal{E}_k\}$, and let $\mathcal{M} \otimes \sigma = \mathcal{M} \otimes \mathcal{E}_1 \otimes \cdots \otimes \mathcal{E}_k$ for a model $\mathcal{M}$.

Corollary 3 *For any model $\mathcal{M}$ and any sequences σ_1, σ_2 of propositional, almost-mutex events with transitive accessibility relations, if $\mathrm{Set}(\sigma_1) = \mathrm{Set}(\sigma_2)$ then $\mathcal{M} \otimes \sigma_1 \leftrightarrows \mathcal{M} \otimes \sigma_2$.*

Proof. Follows immediately from Propositions 1 and 2. □

Proposition 4 *Let σ by any sequence of events from $E = \{\mathcal{N}_\mathrm{t}, \mathcal{N}_\mathrm{g}, \mathcal{B}_\mathrm{t}, \mathcal{B}_\mathrm{g}, \mathcal{F}\}$ Then $\lfloor \mathcal{I} \otimes \sigma \rfloor \leq 6$.*

Proof (Sketch). Using Corollary 3. First note that $\mathcal{F} \otimes \mathcal{E} \leftrightarrows \mathcal{F}$ for any $\mathcal{E} \in E$, so $\lfloor \mathcal{I} \otimes \sigma \rfloor = 1$ for any σ containing $\mathcal{F}$. Also, $\mathcal{N}_\mathrm{t} \otimes \mathcal{N}_\mathrm{g} \leftrightarrows \mathcal{F}$, so the same holds for any sequence containing these two events. Due to symmetry we are left with 6 cases to check: $\sigma \in \{\mathcal{N}_\mathrm{t}, \mathcal{B}_\mathrm{t}, \mathcal{N}_\mathrm{t}\mathcal{B}_\mathrm{t}, \mathcal{N}_\mathrm{t}\mathcal{B}_\mathrm{g}, \mathcal{B}_\mathrm{t}\mathcal{B}_\mathrm{g}, \mathcal{N}_\mathrm{t}\mathcal{B}_\mathrm{t}\mathcal{B}_\mathrm{g}\}$. □

Together with the fact that the bisimulation contraction can be computed in linear time [6], this shows that our toy model indeed stays a toy model. As mentioned above, this may not say much about more realistic models, but guarantees may be found there with similar techniques.

We can also show the fact stated above, saying that we do not need to consider belief revision mechanisms in our simple scenario, since the agents never reach inconsistent belief states (although their beliefs may be mistaken).

Proposition 5 *For any sequence σ of events from E and any agent $i \in \mathcal{A}$, $\mathcal{I} \otimes \sigma \not\models \Box_i \bot$.*

Proof (Sketch). Since $\mathcal{F} \otimes \mathcal{E} \leftrightarrows \mathcal{F}$ for any $\mathcal{E} \in E$ and $\mathcal{I} \otimes \mathcal{F} \not\models \Box_i \bot$, with Corollary 3 we get that $\mathcal{I} \otimes \sigma \not\models \Box_i \bot$ for any σ containing $\mathcal{F}$. Assume there is some σ with $\mathcal{I} \otimes \sigma \models \Box_i \bot$, then there must be no state that i considers possible at the point of $\mathcal{I} \otimes \sigma$. By definition of $\otimes$, the same would then hold for the point of $\mathcal{I} \otimes \sigma \otimes \mathcal{F}$, which is a contradiction. □

We can now proceed to describe the two ways in which we envision such an epistemic model to be used.

3.4 Use case: Scripted behavior

In the original *Thief* game, the behavior of the guard is rigidly connected to particular events. It was proposed in [28, 10] to introduce beliefs as an abstraction layer. For example, if the guard believes that the thief is present, but the guard believes that the thief hasn't noticed the guard, the guard tries to ambush the thief. Otherwise, if the guard believes the thief may have noticed him, but that the thief thinks that he (the guard) in return hasn't noticed him (the thief), the guard tries to exploit that mistaken belief and trick the thief, maybe by playing stupid and suddenly launching a surprise attack. If all else fails, he rushes and attacking him openly. These rules constitute the guard control program in our example scenario, shown in Listing 1.1.

When scripting the guard's behavior, the programmer need not worry about how exactly the beliefs came about, he simply uses the familiar concept of belief in order to express the rules on a high level. As the guard's control script is executed, the knowledge module is queried and determines the truth value of any given formula.

This approach removes from the scripter the burden of having to decide exactly which events cause what, and facilitates adjustments and more complex

```
if B(g, p_t):
    if B(g, not B(t, p_g)):
        g.ambush(t)
    elif B(g, not B(t, B(g, p_t}))):
        g.trick(t)
    else:
        g.rush(t)
```

Listing 1.1: Pseudo-code for a guard in Thief: The Dark Project. `B(i,`φ`)` stands for $\Box_i\varphi$, `g` stands for guard and `t` for thief.

dependencies. For example, the model may be refined so that the noise of an arrow induces the guard to believe that a thief is there and has noticed him, while an arguably more innocent noise such as breaking a twig may induce the guard to believe that a thief is there and hasn't noticed him (he accidentally caused that noise). Only the latter case corresponds to our event n_{t}, and the resulting beliefs would then trigger the behavior rule that says to try an ambush. The point is that this belief model is independent from the behavior and can be tested and tuned separately.

3.5 Use case: Autonomous planning

Ideally, handcrafted rules would be as few as possible, and the artificial agents would act largely autonomously, including proactively trying to bring about desirable (epistemic) situations. We discuss how a planning guard may be realized in our simple scenario, before we set out to propose a general DEL planning framework in § 4.

First, we specify which of the events we discussed can be brought about by the guard, and thus should be in his repertoire of actions. For now, we simply assume that the guard can make a noise (n_{g}), he can let himself be seen from behind (b_{t}; for example, by walking around with his head turned towards the wall), and he can step out and provoke a face-to-face encounter (f). These actions are associated with the corresponding DEL events from Figure 4, which also encapsulate the actions' preconditions. For example, f can only happen if both are present, and correspondingly, the guard only knows that he can make f happen if he knows that both are present.

Besides these actions corresponding to our DEL events, we consider the following additional actions specifying the different attacks described in § 3.4:

- ambush, with precondition $\mathsf{pre}(\mathsf{ambush}) = p_{\mathrm{t}} \wedge \neg\Box_{\mathrm{t}} p_{\mathrm{g}}$
- trick, with precondition $\mathsf{pre}(\mathsf{trick}) = p_{\mathrm{t}} \wedge \Box_{\mathrm{t}} p_{\mathrm{g}} \wedge \neg\Box_{\mathrm{t}}\Box_{\mathrm{g}} p_{\mathrm{t}}$
- rush, with precondition $\mathsf{pre}(\mathsf{rush}) = p_{\mathrm{t}}$.

Plans can be assessed as to whether they are legal, believed by the guard to be legal, or believed *to be believed* to be legal at each intermediate point of the

plan (i.e., that plan can be "knowingly" executed). Of course, the guard agent should only be able to access the last two assessments.

For example, two potential plans would be $P_1 = b_t, \mathsf{trick}$ and $P_2 = n_g, \mathsf{ambush}$. In the initial situation $\mathcal{I}$, P_1 is legal, but the guard does not know this since he is not sure of the thief's presence. After seeing the thief from behind, the guard does take P_1 to be legal, although it is not the case that he would believe it at each point of the plan. Formally, by slight abuse of notation, we have $\mathcal{I} \models \langle \mathcal{B}_t \rangle \langle \mathsf{trick} \rangle \top$ but $\mathcal{I} \not\models \Box_g \langle \mathcal{B}_t \rangle \langle \mathsf{trick} \rangle \top$, and $\mathcal{I} \otimes \mathcal{B}_g \models \Box_g \langle \mathcal{B}_t \rangle \langle \mathsf{trick} \rangle \top$ but $\mathcal{I} \otimes \mathcal{B}_g \not\models \Box_g \langle \mathcal{B}_t \rangle \Box_g \langle \mathsf{trick} \rangle \top$. This last thing is due to the fact that we modeled being seen from behind as something that one never assumes to happen. In that sense it is not an action the guard can knowingly perform—it would be more appropriate to formalize such an action as DEL event which does get reflected in the guard's epistemic state, but which looks to the thief like $\mathcal{B}_t$. Due to the modular nature of DEL, such issues can be taken care of simply by modifying or adding the affected event models.

With P_2, we have $\mathcal{I} \not\models \langle \mathcal{N}_g \rangle \langle \mathsf{ambush} \rangle \top$ along with $\mathcal{I} \not\models \Box_g \langle \mathcal{N}_g \rangle \langle \mathsf{ambush} \rangle \top$ and $\mathcal{I} \not\models \Box_g \langle \mathcal{N}_g \rangle \Box_g \langle \mathsf{ambush} \rangle \top$. In fact the same is true in any situation, since by the very act of making noise, the guard destroys the precondition for an ambush.

We have implemented a naive planner which considers all plans up to a certain length ending with one of the attack actions, and queries the knowledge module as to their (current) feasibility. One research goal is to find shortcuts for pruning the space of potential plans to a manageable size. For example, we can note that positive formulas are persistent, so any plan under construction can be discarded as soon as a state is reached where a negated positive goal formula is violated. Once the thief does know that the guard has noticed him (violating the last conjunct in $\mathsf{pre}(\mathsf{trick})$), no further events can destroy that knowledge and the search for any plan ending with trick can be aborted.

We end the discussion of planning in our scenario by showing the protocol of an example run.

```
The scene starts.
Selected plan: None.
Guard sees thief from behind.
Selected plan: n_g,trick         Avoid: n_t f
Something happens which is not part of the plan:
Thief steps on a twig or makes some other noise (n_t)
Selected plan: ambush            Avoid: b_t n_g f
```

Note that this guard, other than the one from Listing 1.1, prefers to trick the thief and only falls back to the plan of ambushing once tricking is not possible anymore. Also, our simple planner produces a list of events that should be avoided in order to keep the selected plan (believed to be) legal. In that sense, it produces compositional behavior: Ambush is just an attack while avoiding being seen or making noise; trick is an attack avoiding a face-to-face encounter or the thief making noise (thus losing his belief that the guard hasn't noticed him). Of course, it is not within the guard's power to prevent that last event.

4 Towards a general framework for DEL planning

DEL has established itself as a standard conceptual model for epistemic situations and change, allowing to represent arbitrary events, including an algorithmic definition of how the events affect the epistemic models. Together with its conceptual clarity, this makes DEL a promising framework for knowledge- or belief-based parts of computer games. Making planning into a crucial element of narrative design for computer games has been proposed by Riedl and Young [15, 16]. Combining these aspects, we here propose to study a general framework of DEL planning.

The classic *planning problem* consists of a description of the world, the agent's goal and a description of the possible actions in some appropriate formal language. A planning algorithm consists of sequences of possible actions which when executed will achieve the goal.

To attempt an initial definition of a DEL planning problem, we fix a pointed epistemic model $(\mathcal{M}, w)$ and a finite set $\mathfrak{E}$ of pointed event models. As mentioned at the end of § 2, we consider the tree LS of legal sequences σ from $\mathfrak{E}$ as our space of possible plans, and we allow to impose additional rules on when events can occur in the form of specifying a subtree $T \subseteq \mathrm{LS}$.

Definition 6 (Absolute DEL planning problem) *Given $(\mathcal{M}, w)$, $\mathfrak{E}$, a subtree $T \subseteq \mathrm{LS}$, and a formula $\varphi \in \mathcal{L}_{\mathrm{DEL}}$, produce a sequence $\sigma \in T$ such that $\mathcal{M} \otimes \sigma \models \varphi$.*

The absolute DEL planning problem does not talk about agents and whether the plan is realizable by a single agent. In fact, the DEL formalism as described in § 2 does not talk about whether events are performed as actions of agents, and of which agents. We have already seen in our *Thief* example that in practical applications, it will matter a lot whether certain events are within the power of a particular agent. In order to represent this, we consider a function power : $\mathcal{A} \to \wp(\mathfrak{E})$ that tells us which events an agent can bring about. Here, if $i \in \mathcal{A}$, we interpret $\mathcal{E} \in \mathrm{power}(i)$ as "agent i can perform action $\mathcal{E}$". If σ is a sequence of actions, we write $\sigma \in \mathrm{power}(i)$ if for each $\mathcal{E} \in \mathrm{Set}(\sigma)$, we have $\mathcal{E} \in \mathrm{power}(i)$. Similarly, for a partial sequence, i.e., a partial function $\widehat{\sigma} : \{0, \ldots, N\} \to \mathfrak{E}$ (for some N), we write $\widehat{\sigma} \in \mathrm{power}(i)$, if for all $n \in \mathrm{dom}(\widehat{\sigma})$, we have $\widehat{\sigma}(n) \in \mathrm{power}(i)$.[8]

Definition 7 (Single-agent DEL planning problem) *Given $(\mathcal{M}, w)$, $\mathfrak{E}$, a subtree $T \subseteq \mathrm{LS}$, a function* power*, an agent $i \in \mathcal{A}$, and a formula $\varphi \in \mathcal{L}_{\mathrm{DEL}}$, produce a sequence $\sigma \in \mathrm{LS}$ such that $\mathcal{M} \otimes \sigma \models \varphi$ and $\sigma \in \mathrm{power}(i)$.*

The single-agent DEL planning problem is asking for a plan that can be executed by the agent alone, assuming no (interfering) actions of other agents. More interesting (but also harder) is therefore the following problem that essentially asks for a *winning strategy* in the usual game-theoretic sense.

[8] These definitions are easily extended to sets of agents if we want to include actions that can only jointly be performed by a group of agents.

Definition 8 (Adversarial DEL planning problem) *Given* $(\mathcal{M}, w)$*,* $\mathfrak{E}$*, a subtree* $T \subseteq \mathrm{LS}$*, a function* power*, an agent* $i \in \mathcal{A}$*, and a formula* $\varphi \in \mathcal{L}_{\mathrm{DEL}}$*, produce a partial function* $\widehat{\sigma} : \{0, \ldots, N\} \to \mathfrak{E}$ *such that*

1. $\widehat{\sigma} \in \mathrm{power}(i)$*,*
2. *there is a sequence* $\sigma \in T$ *such that* $\widehat{\sigma} \subseteq \sigma$*, and*
3. *for all sequences* $\sigma \in T$ *with* $\widehat{\sigma} \subseteq \sigma$*, we have* $\mathcal{M} \otimes \sigma \models \varphi$*.*

Note that this formulation assumes a kind of worst-case scenario, or in game-theoretic terms, a zero-sum game. Ultimately, we would like to depart from the extremes embodied in the previous two definitions (assuming no interference versus assuming complete opposition) and study a general **strategic DEL planning problem** where agents take (interfering or cooperative) actions of third agents into account, and may to some extent try to anticipate them. However, while there exists work in the context of DEL on defining the information content of events in strategic settings [8] and modeling goals and preferences [21], things are far less clear in such a case, and we therefore leave it to future work.

5 Discussion and Future Work

We gave the our discussion of §§ 3 and 4 in the framework of DEL. Of course, the most interesting applications are not about knowledge, but rather about belief, so a first step would be to phrase the planning problems of this paper in *dynamic doxastic logic* instead of DEL. There are a number of simplifications we made in our set-up that could be removed: we did not consider the possibility of events that change the valuation function (cf. [22]) nor the possibility that an agent's ability to perform an action may change over time. Generalizing our framework to include these possibilities would be a natural next step.

Of course, the main technical issue is that in general it may be infeasible to solve the DEL planning problem. The search space can easily be infinite, and even if it is finite, it can be very large. Therefore, two crucial topics for further research will be the following:

- Examine the long-term expansion of models under iterated updates (related to [18], but we are foremost interested in finite models) and identify natural and general classes of actions that allow arguments such as our crucial Proposition 4 in § 3.
- Find compact representations of models (cf. techniques from model checking [5]) on which the product operation can operate directly.

Finally there is the large and mature field of "classic" planning with many effective and proven optimization strategies and other techniques. Some of these may be directly applicable to DEL planning or inspire suitable variants.

References

1. T. Ågotnes and H. van Ditmarsch. What will they say?—Public announcement games, 2009. Paper presented at *Logic, Game Theory and Social Choice 6* in Tsukuba, Japan.
2. G. Aucher. *Perspectives on Belief and Change*. PhD thesis, Université de Toulouse and University of Otago, 2008.
3. A. Baltag, L. Moss, and S. Solecki. The logic of public announcements, common knowledge and private suspicions. In I. Gilboa, editor, *Proceedings of the 7th Conference on Theoretical Aspects of Rationality and Knowledge (TARK-98), Evanston, IL, USA, July 22-24, 1998*, pages 43–56. Morgan Kaufmann, 1998.
4. A. Baltag and S. Smets. A qualitative theory of dynamic interactive belief revision. In G. Bonanno, W. van der Hoek, and M. Wooldridge, editors, *Logic and the Foundations of Game and Decision Theory (LOFT 7)*, volume 3 of *Texts in Logic and Games*, pages 9–58. Amsterdam University Press, Amsterdam, 2008.
5. E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. MIT Press, 1999.
6. A. Dovier, C. Piazza, and A. Policriti. A fast bisimulation algorithm. In G. Berry, H. Comon, and A. Finkel, editors, *Computer Aided Verification. 13th International Conference, CAV 2001, Paris, France, July 18-22, 2001. Proceedings*, volume 2102 of *Lecture Notes in Computer Science*, pages 79–90. Springer, 2001.
7. R. Fagin, J. Halpern, Y. Moses, and M. Vardi. *Reasoning about Knowledge*. The MIT Press, Boston, 1995.
8. J. Gerbrandy. Communication strategies in games. *Journal of Applied Non-Classical Logics*, 17(2):197–211, 2007.
9. F. Giunchiglia and P. Traverso. Planning as model checking. In S. Biundo and M. Fox, editors, *Recent Advances in AI Planning. 5th European Conference on Planning, ECP'99. Durham, UK, September 8-10, 1999. Proceedings.*, volume 1809 of *Lecture Notes in Artificial Intelligence*, pages 1–20, Berlin, 2000. Springer.
10. E. Kennerly, A. Witzel, and J. A. Zvesper. Thief belief. In B. Löwe, editor, *LSIR-2: Logic and the Simulation of Interaction and Reasoning Workshop at IJCAI-09*, pages 47–51, Pasadena, CA, 2009.
11. B. Löwe and E. Pacuit. An abstract approach to reasoning about games with mistaken and changing beliefs. *Australasian Journal of Logic*, 6:162–181, 2008.
12. B. Löwe, E. Pacuit, and S. Saraf. Identifying the structure of a narrative via an agent-based logic of preferences and beliefs: Formalizations of episodes from *CSI: Crime Scene Investigation*$^{\text{TM}}$. In M. Duvigneau and D. Moldt, editors, *MOCA'09. Fifth International Workshop on Modelling of Objects, Components, and Agents, Hamburg 2009*, 2009.
13. J. C. Meyer and W. van der Hoek. *Epistemic Logic for AI and Computer Science*. Cambridge University Press, 1995.
14. G. R. Renardel de Lavalette and H. van Ditmarsch. Epistemic actions and minimal models, 2002. Available on the authors' websites.
15. M. O. Riedl and R. M. Young. Character-focused narrative planning. Unpublished manuscript, 2003.
16. M. O. Riedl and R. M. Young. From linear story generation to branching story graphs. *IEEE Computer Graphics and Applications*, 26(3):23–31, 2006.
17. J. Ruan. *Reasoning about Time, Action and Knowledge in Multi-Agent Systems*. PhD thesis, University of Liverpool, 2008.
18. T. Sadzik. Exploring the iterated update universe, 2006. ILLC Publications PP-2006-26.

19. J. van Benthem. One is a lonely number: on the logic of communication. In Z. Chatzidakis, P. Koepke, and W. Pohlers, editors, *Logic Colloquium '02. Joint proceedings of the Annual European Summer Meeting of the Association for Symbolic Logic and the Biannual Meeting of the German Association for Mathematical Logic and the Foundations of Exact Sciences (the Colloquium Logicum) held in Münster, August 3–11, 2002*, volume 27 of *Lecture Notes in Logic*. Association for Symbolic Logic, 2006.
20. J. van Benthem, J. Gerbrandy, T. Hoshi, and E. Pacuit. Merging frameworks for interaction. *Journal of Philosophical Logic*, 38(5):491–526, 2009.
21. J. van Benthem and F. Liu. Dynamic logic of preference upgrade. *Journal of Applied Non-Classical Logic*, 17(2):157–182, 2007.
22. J. van Benthem, J. van Eijck, and B. Kooi. Logics of communication and change. *Information and Computation*, 204(11):1620–1662, 2006.
23. W. van der Hoek and M. Wooldridge. Tractable multiagent planning for epistemic goals. In C. Castelfranchi, editor, *The First International Joint Conference on Autonomous Agents & Multiagent Systems, AAMAS 2002, July 15-19, 2002, Bologna, Italy, Proceedings*, pages 1167–1174, Bologna, Italy, 2002. ACM.
24. H. van Ditmarsch. *Knowledge Games*. PhD thesis, Groningen University, 2000. ILLC Publications DS-2000-06.
25. H. van Ditmarsch, W. van der Hoek, and B. Kooi. *Dynamic Epistemic Logic*, volume 337 of *Synthese Library*. Springer, 2008.
26. J. van Eijck. DEMO—a demo of epistemic modelling. In J. van Benthem, D. Gabbay, and B. Löwe, editors, *Interactive Logic. Selected Papers from the 7th Augustus de Morgan Workshop, London*, volume 1 of *Texts in Logic and Games*, pages 303–362, 2007.
27. J. van Eijck, J. Ruan, and T. Sadzik. Action emulation. Draft paper, 2008.
28. A. Witzel, J. Zvesper, and E. Kennerly. Explicit knowledge programming for computer games. In C. Darken and M. Mateas, editors, *Proceedings of the Fourth Artificial Intelligence and Interactive Digital Entertainment Conference, October 22-24, 2008, Stanford, California, USA*. AAAI Press, 2008.