Symbolic Model Checking for Dynamic Epistemic Logic – S5 and Beyond

van Benthem, J.; van Eijck, J.; Gattinger, M.; Su, K.

Published in:
Journal of Logic and Computation

DOI:
10.1093/logcom/exx038

Symbolic Model Checking for Dynamic Epistemic Logic — S5 and Beyond

Johan van Benthem[1,2,3], Jan van Eijck[1,4], Malvin Gattinger[1], and Kaile Su[5,6]

[1] Institute for Logic, Language & Computation, University of Amsterdam
[2] Department of Philosophy, Stanford University
[3] Changjiang scholars program, Tsinghua University
[4] Centrum Wiskunde & Informatica, Amsterdam
[5] Institute for Integrated and Intelligent Systems, Griffith University
[6] Department of Computer Science, Jinan University

Abstract. Dynamic Epistemic Logic (DEL) can model complex information scenarios in a way that appeals to logicians. However, existing DEL implementations are ad-hoc, so we do not know how the framework really performs. For this purpose, we want to hook up with the best available model-checking and SAT techniques in computational logic. We do this by first providing a bridge: a new faithful representation of DEL models as so-called knowledge structures that allow for symbolic model checking. For more complex epistemic change we introduce knowledge transformers analogous to action models. Next, we show that we can now solve well-known benchmark problems in epistemic scenarios much faster than with existing methods for DEL. We also compare our approach to model checking for temporal logics. Finally, we show that our method is not just a matter of implementation, but that it raises significant issues about logical representation and update.[7]

1 Introduction

We bring together two strains in the area of epistemic model checking. On one side, there are many frameworks for symbolic model checking on interpreted systems using temporal logics [31,38]. On the other hand, there are explicit model checkers for variants of Dynamic Epistemic Logic (DEL) like DEMO [18] and the optimized successor DEMO-S5 [19]. The latter provide superior usability as they allow specification in dynamic languages directly, but inferior performance. This reflects that the cradle of DEL was logic and philosophy, not computer science: Models are just abstract mathematical objects whose size does not matter. To actually implement model checking however, we need to think about concrete data structures. Implementing the standard logical semantics means that we explicitly spell out all states of a Kripke model. Already for toy examples like the muddy children the number of states is exponential in the number of agents

[7] An earlier version [3] of this paper appeared in the proceedings of LORI-V. The main additions and changes here are the generalization to non-S5 logics and a detailed explanation of knowledge transformers.

and propositions. Solutions to this state explosion problem have been found for temporal logics. The goal of our work is therefore to connect the two worlds of symbolic model checking and DEL in order to gain new insights on both sides.

Existing work on model checking DEL mainly focuses on specific examples, for example the Dining Cryptographers [35], the Sum and Product riddle [33] or Russian Cards [15]. Given these specific approaches, a general approach to symbolic model checking the full DEL language is desirable. A first step is [38] which presents symbolic model checking for temporal logics of knowledge. However, it does not cover announcements or other dynamics. The framework here extends these ideas with dynamic operators and a twist on the semantics.

Our knowledge structures are similar in spirit to hypercubes from [32], but of a different type: We do not use interpreted systems and temporal relations are not part of our models. Hence also our language does not contain temporal operators but primitives for epistemic events like announcements.

Related to our work is also [16] where DEL is translated into temporal epistemic logics for which symbolic model checkers exist. However, this method has not been implemented and the complexity and performance are not known. We do not translate to a temporal logic but check DEL formulas directly.

The paper is structured as follows. In Section 2 we recall standard semantics of DEL as in [14]. We then present knowledge structures in Section 3 and explain them in detail using the famous Muddy Children example in Section 4. In Section 5 we discuss some details of the implementation and presents benchmark results for various examples. Our main theoretical results are in Section 6 where we use translations to prove that S5 Kripke models and knowledge structures are equally expressive. Sections 7 and 8 extend our framework to action models from [2] and non-S5 logics, respectively. Section 9 gives a conclusion and suggestions for further research. An Appendix presents the implementation.

All source code of our model checker, technical documentation and a simple web interface can be found at `https://github.com/jrclogic/SMCDEL`.

2 Dynamic Epistemic Logic on Kripke Models

Definition 1. *Fix a set of propositions V and a finite set of agents I. The DEL language $\mathcal{L}(V)$ is given by*

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_i\varphi \mid C_\Delta\varphi \mid [\varphi]\varphi \mid [\varphi]_\Delta\varphi$$

where $p \in V$, $i \in I$ and $\Delta \subseteq I$. We also use the abbreviations $\varphi \vee \psi := \neg(\neg\varphi \wedge \neg\psi)$ and $\varphi \to \psi := \neg(\varphi \wedge \neg\psi)$. The boolean *formulas are $\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi$.*

The formula $K_i\varphi$ is read as *"agent i knows φ"* while $C_\Delta\varphi$ says that φ is common knowledge among agents in Δ. The formula $[\psi]\varphi$ indicates that after a *public announcement* of ψ, φ holds. In contrast, $[\psi]_\Delta\varphi$ says that after announcing ψ to the agents in Δ, φ holds. If $\Delta = \{i\}$ for a single agent $i \in I$, then we also write i instead of $\{i\}$. The standard semantics for $\mathcal{L}(V)$ are given by means of Kripke models as follows.

Definition 2. *A* Kripke model *for a set of agents $I = \{1, \ldots, n\}$ is a tuple $\mathcal{M} = (W, \pi, \mathcal{K}_1, \ldots, \mathcal{K}_n)$, where W is a set of* worlds*, π associates with each world a truth assignment to the primitive propositions, so that $\pi(w)(p) \in \{\top, \bot\}$ for each world w and primitive proposition p, and $\mathcal{K}_1, \ldots, \mathcal{K}_n$ are binary accessibility relations on W. By convention, $W^{\mathcal{M}}$, $\mathcal{K}_i^{\mathcal{M}}$ and $\pi^{\mathcal{M}}$ are used to refer to the components of $\mathcal{M}$. We omit the superscript $\mathcal{M}$ if it is clear from context. Finally, let $\mathcal{C}_{\Delta}^{\mathcal{M}}$ be the transitive closure of $\bigcup_{i \in \Delta} \mathcal{K}_i^{\mathcal{M}}$.*

A pointed Kripke model *is a pair $(\mathcal{M}, w)$ consisting of a Kripke model and a world $w \in W^{\mathcal{M}}$. A model $\mathcal{M}$ is called an* S5 Kripke model *iff, for every i, $\mathcal{K}_i^{\mathcal{M}}$ is an equivalence relation. A model $\mathcal{M}$ is called* finite *iff $W^{\mathcal{M}}$ is finite.*

Definition 3. *Semantics for $\mathcal{L}(V)$* on pointed Kripke models *are given inductively as follows.*

1. $(\mathcal{M}, w) \vDash p$ *iff* $\pi^M(w)(p) = \top$.
2. $(\mathcal{M}, w) \vDash \neg\varphi$ *iff not* $(\mathcal{M}, w) \vDash \varphi$
3. $(\mathcal{M}, w) \vDash \varphi \wedge \psi$ *iff* $(\mathcal{M}, w) \vDash \varphi$ *and* $(\mathcal{M}, w) \vDash \psi$
4. $(\mathcal{M}, w) \vDash K_i\varphi$ *iff for all* $w' \in W$*, if* $w\mathcal{K}_i^M w'$*, then* $(\mathcal{M}, w') \vDash \varphi$.
5. $(\mathcal{M}, w) \vDash C_{\Delta}\varphi$ *iff for all* $w' \in W$*, if* $w\mathcal{C}_{\Delta}^{\mathcal{M}} w'$*, then* $(\mathcal{M}, w') \vDash \varphi$.
6. $(\mathcal{M}, w) \vDash [\psi]\varphi$ *iff* $(\mathcal{M}, w) \vDash \psi$ *implies* $(\mathcal{M}^{\psi}, w) \vDash \varphi$ *where $\mathcal{M}^{\psi}$ is a new Kripke model defined by the set* $W^{\mathcal{M}^{\psi}} := \{w \in W^{\mathcal{M}} \mid (\mathcal{M}, w) \vDash \psi\}$*, the relations* $\mathcal{K}_i^{\mathcal{M}^{\psi}} := \mathcal{K}_i^M \cap \left(W^{\mathcal{M}^{\psi}}\right)^2$ *and the valuation* $\pi^{\mathcal{M}^{\psi}}(w) := \pi^{\mathcal{M}}(w)$.
7. $(\mathcal{M}, w) \vDash [\psi]_{\Delta}\varphi$ *iff* $(\mathcal{M}, w) \vDash \psi$ *implies that* $(\mathcal{M}_{\psi}^{\Delta}, w) \vDash \varphi$ *where* $(\mathcal{M}_{\psi}^{\Delta}, w)$ *is a new Kripke model defined by the same set of worlds* $W^{\mathcal{M}_{\psi}^{\Delta}} := W^{\mathcal{M}}$*, modified relations such that*
   - *if $i \in \Delta$, let $w\mathcal{K}_i^{\mathcal{M}_{\psi}^{\Delta}} w'$ iff (i) $w\mathcal{K}_i^{\mathcal{M}} w'$ and (ii) $(\mathcal{M}, w) \vDash \psi$ iff $(\mathcal{M}, w') \vDash \psi$*
   - *otherwise, let $w\mathcal{K}_i^{\mathcal{M}_{\psi}^{\Delta}} w'$ iff $w\mathcal{K}_i^{\mathcal{M}} w'$*

   and the same valuation $\pi^{\mathcal{M}_{\psi}^{\Delta}}(w) := \pi^{\mathcal{M}}(w)$.

Definition 3 is standard and well-known, up to the last part which describes a semi-private group announcement: $[\psi]_{\Delta}\varphi$ expresses that φ holds after it was truthfully announced to the agents in Δ that ψ holds. We interpret it by cutting the links between worlds that disagree on ψ for all agents in Δ. This announcement is *not secret*: Other agents learn that agents in Δ learn whether ψ. To illustrate this, consider the following example.

Example 1. Alice has applied for a post-doc position and Bob knows this. A messenger enters and gives Alice an envelope with the university logo on it. She reads the letter and learns that she got the position. Bob however only learns that Alice learns *whether* she got the position.

Let p stand for "Alice gets the position." and consider the initial model $\mathcal{M}$ depicted in Figure 1 where both Alice and Bob do not know whether p. We formalize Alice reading the letter as $[p]_{\text{Alice}}$ and model $\mathcal{M}_p^{\text{Alice}}$ is the result of this update. Again note that Bob himself does not learn whether p, but he does learn that Alice learns whether p.

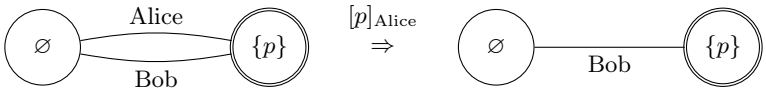


Fig. 1. Alice reading the letter as a semi-private announcement.

We include this semi-private announcement here already to show how our symbolic representation in the next section captures more than just public announcements. More complex epistemic actions are covered later in Section 7 and we generalize our methods to non-S5 phenomena in Section 8, including truly private announcements in which the agents in Δ do not learn anything.

3 Knowledge Structures

While the preceding semantics is standard in logic, it cannot serve directly as an input to current sophisticated model-checking techniques. For this purpose, in this section we introduce a new format, *knowledge structures*. Their main advantage is that also knowledge and results of announcements can be computed via purely boolean operations. We first recapitulate some notions and abbreviations.

Given a set of propositional variables P, we identify a *truth assignment over* P with a subset of P. We say a formula φ is a formula *over* P if each propositional variable occurring in φ is in P. For convenience, we use the logical constants $\top$ and $\bot$ which are always true and always false, respectively. We also use $\models$ to denote the usual satisfaction relation between a truth assignment and a formula.

We use substitution and quantification as follows. For any formula φ and $\psi \in \{\top, \bot\}$, and any propositional variable p, let $\varphi(\frac{p}{\psi})$ denote the result of replacing every p in φ by ψ. For any $A = \{p_1, \ldots, p_n\}$, let $\varphi(\frac{A}{\psi}) := \psi(\frac{p_1}{\psi})(\frac{p_2}{\psi}) \ldots (\frac{p_n}{\psi})$, i.e. the result of substituting ψ for all elements of A. We use $\forall p\varphi$ to denote $\varphi\left(\frac{p}{\top}\right) \wedge \varphi\left(\frac{p}{\bot}\right)$. For any $A = \{p_1, \ldots, p_n\}$, let $\forall A\varphi := \forall p_1 \forall p_2 \ldots \forall p_n \varphi$.

Definition 4. *Suppose we have n agents. A* knowledge structure *is a tuple $\mathcal{F} = (V, \theta, O_1, \ldots, O_n)$ where V is a finite set of propositional variables, θ is a boolean formula over V and for each agent i, $O_i \subseteq V$.*

Set V is the vocabulary *of $\mathcal{F}$. Formula θ is the* state law *of $\mathcal{F}$. It determines the set of states of $\mathcal{F}$ and may only contain boolean operators. The variables in O_i are called agent i's* observable variables*. An assignment over V that satisfies θ is called a* state *of $\mathcal{F}$. Any knowledge structure only has finitely many states. Given a state s of $\mathcal{F}$, we say that $(\mathcal{F}, s)$ is a* scene *and define the* local state *of an agent i at s as $s \cap O_i$.*

To interpret common knowledge we use the following definitions. Given a knowledge structure $(V, \theta, O_1, \ldots, O_n)$ and a set of agents Δ, let $\mathcal{E}_\Delta$ be the relation on states of $\mathcal{F}$ defined by $(s, t) \in \mathcal{E}_\Delta$ iff there exists an $i \in \Delta$ with $s \cap O_i = t \cap O_i$. and let $\mathcal{E}^*_\mathcal{V}$ to denote the transitive closure of $\mathcal{E}_\mathcal{V}$.

Example 2. Consider the knowledge structure

$$\mathcal{F} := (V = \{p, q\}, \theta = p \to q, O_1 = \{p\}, O_2 = \{q\})$$

Here the vocabulary consists of two propositions. The state law is $p \to q$, hence the states of $\mathcal{F}$ are the three assignments satisfying that formula. To simplify notation we write assignments as the set of propositions they make true. The states of $\mathcal{F}$ are thus $\varnothing$, $\{q\}$ and $\{p, q\}$. Moreover, $\mathcal{F}$ describes two agents who each observe one of the propositions. Intuitively, this can be understood as information about knowing *whether*: Agent 1 knows whether p is true and agent 2 knows whether q is true. We also use this knowledge structure in Example 3 below and compute an equivalent Kripke model in Example 6.

We now give alternative semantics for $\mathcal{L}(V)$ on knowledge structures. Definitions 5 and 6 run in parallel, both proceeding by the structure of φ.

Definition 5. *Semantics for* $\mathcal{L}(V)$ on scenes *are defined inductively as follows.*

1. $(\mathcal{F}, s) \vDash p$ *iff* $s \vDash p$.
2. $(\mathcal{F}, s) \vDash \neg\varphi$ *iff not* $(\mathcal{F}, s) \vDash \varphi$
3. $(\mathcal{F}, s) \vDash \varphi \wedge \psi$ *iff* $(\mathcal{F}, s) \vDash \varphi$ *and* $(\mathcal{F}, s) \vDash \psi$
4. $(\mathcal{F}, s) \vDash K_i\varphi$ *iff for all* t *of* $\mathcal{F}$, *if* $s \cap O_i = t \cap O_i$, *then* $(\mathcal{F}, t) \vDash \varphi$.
5. $(\mathcal{F}, s) \vDash C_\Delta\varphi$ *iff for all* t *of* $\mathcal{F}$, *if* $(s, t) \in \mathcal{E}^*_\Delta$, *then* $(\mathcal{F}, t) \vDash \varphi$.
6. $(\mathcal{F}, s) \vDash [\psi]\varphi$ *iff* $(\mathcal{F}, s) \vDash \psi$ *implies* $(\mathcal{F}^\psi, s) \vDash \varphi$ *where* $\|\psi\|_\mathcal{F}$ *is given by Definition 6 and*
$$\mathcal{F}^\psi := (V, \theta \wedge \|\psi\|_\mathcal{F}, O_1, \dots, O_n)$$
7. $(\mathcal{F}, s) \vDash [\psi]_\Delta\varphi$ *iff* $(\mathcal{F}, s) \vDash \psi$ *implies* $(\mathcal{F}^\Delta_\psi, s \cup \{p_\psi\}) \vDash \varphi$ *where* p_ψ *is a new propositional variable,* $\|\psi\|_\mathcal{F}$ *is a boolean formula given by Definition 6 and*
$$\mathcal{F}^\Delta_\psi := (V \cup \{p_\psi\}, \theta \wedge (p_\psi \leftrightarrow \|\psi\|_\mathcal{F}), O^*_1, \dots, O^*_n)$$
where $O^*_i := \begin{cases} O_i \cup \{p_\psi\} & \text{if } i \in \Delta \\ O_i & \text{otherwise} \end{cases}$

If we have $(\mathcal{F}, s) \vDash \varphi$ *for all states* s *of* $\mathcal{F}$, *then we say that* φ *is valid on* $\mathcal{F}$ *and write* $\mathcal{F} \vDash \varphi$.

Before defining the boolean equivalents of formulas, we can already explain some similarities and differences between Definitions 3 and 5. The semantics of the boolean connectives are the same. For the knowledge operators, on Kripke models we use an accessibility relation $\mathcal{K}_i$. On knowledge structures this is replaced with the condition $s \cap O_i = t \cap O_i$, inducing an equivalence relation on the states. We can already guess that knowledge structures encode S5 Kripke models.

Example 3. Consider again the knowledge structure $\mathcal{F}$ from Example 2. We can easily check that $(\mathcal{F}, \varnothing) \vDash K_1\neg p$ holds: The only states t of $\mathcal{F}$ such that $\varnothing \cap O_1 = t \cap O_1$ are $\varnothing$ and $\{q\}$, and we have $(\mathcal{F}, \varnothing) \vDash \neg p$ and $(\mathcal{F}, \{q\}) \vDash \neg p$.

Similarly we can check that $(\mathcal{F}, \{p,q\}) \vDash K_1 q$: There is no other state $t \neq \{p,q\}$ such that $\{p,q\} \cap O_1 = t \cap O_1$ because the state law $\theta = p \to q$ rules out $\{p\}$. Intuitively, even though agent 1 does not observe q, she observes p and the state law $p \to q$ implies that q also must be true. In general, the state law of a knowledge structure is always valid on it and therefore also common knowledge among all agents. In this case: $\mathcal{F} \vDash C_{\{1,2\}}(p \to q)$.

Our intuitive understanding of observational variables as knowing whether can now also be stated formally: $\mathcal{F} \vDash K_1 p \vee K_1 \neg p$ and $\mathcal{F} \vDash K_2 q \vee K_2 \neg q$.

The following definition of local boolean equivalents is the crucial ingredient that enables symbolic model checking on our structures.

Definition 6. *For any knowledge structure $\mathcal{F} = (V, \theta, O_1, \ldots, O_n)$ and any formula φ we define its* local boolean translation $\|\varphi\|_{\mathcal{F}}$ *as follows.*

1. *For any primitive formula, let $\|p\|_{\mathcal{F}} := p$.*
2. *For negation, let $\|\neg\psi\|_{\mathcal{F}} := \neg\|\psi\|_{\mathcal{F}}$.*
3. *For conjunction, let $\|\psi_1 \wedge \psi_2\|_{\mathcal{F}} := \|\psi_1\|_{\mathcal{F}} \wedge \|\psi_2\|_{\mathcal{F}}$.*
4. *For knowledge, let $\|K_i\psi\|_{\mathcal{F}} := \forall(V \setminus O_i)(\theta \to \|\psi\|_{\mathcal{F}})$.*
5. *For common knowledge, let $\|C_\Delta\psi\|_{\mathcal{F}} := \mathbf{gfp}\Lambda$ where Λ is the following operator on boolean formulas and $\mathbf{gfp}\Lambda$ denotes its greatest fixed point:*

$$\Lambda(\alpha) := \|\psi\|_{\mathcal{F}} \wedge \bigwedge_{i \in \Delta} \forall(V \setminus O_i)(\theta \to \alpha)$$

6. *For public announcements, let $\|[\psi]\xi\|_{\mathcal{F}} := \|\psi\|_{\mathcal{F}} \to \|\xi\|_{\mathcal{F}^\psi}$.*
7. *For group announcements, let $\|[\psi]_\Delta\xi\|_{\mathcal{F}} := \|\psi\|_{\mathcal{F}} \to (\|\xi\|_{\mathcal{F}^\Delta_\psi})(\frac{p_\psi}{\top})$.*

where $\mathcal{F}^\psi$ and $\mathcal{F}^\Delta_\psi$ are as given by Definition 5.

Example 4. Using the structure $\mathcal{F}$ from Example 2 we have:

$$\begin{aligned}
\|K_2(p \vee q)\|_{\mathcal{F}} &= \forall(V \setminus O_2)(\theta \to \|p \vee q\|_{\mathcal{F}}) \\
&= \forall p((p \to q) \to (p \vee q)) \\
&= ((\top \to q) \to (\top \vee q)) \wedge ((\bot \to q) \to (\bot \vee q)) \\
&= (q \to \top) \wedge (\top \to q) \\
&= q
\end{aligned}$$

One can check that indeed the formulas $K_2(p \vee q)$ and q are true at the same states of $\mathcal{F}$, namely $\{p,q\}$ and $\{q\}$. Note that here we simplified the notation and considered equivalent boolean formulas to be identical, so in particular we do not consider succinctness of DEL formulas and their translations. This is in line with the implementation presented in Section 5 where we will use binary decision diagrams to represent boolean functions.

The next section contains more complex examples of this translation. Here it remains to show that the boolean translations are indeed locally equivalent.

Theorem 1. *Definition 6 preserves and reflects truth. That is, for any formula φ and any scene $(\mathcal{F}, s)$ we have that $(\mathcal{F}, s) \vDash \varphi$ iff $s \vDash \|\varphi\|_{\mathcal{F}}$.*

Proof. By induction on φ. The base case for atomic propositions is immediate. In the induction step, negation and conjunction are standard.

For the case of knowledge, remember how we defined boolean quantification just before Definition 4 and note the following equivalences:

$$
\begin{array}{ll}
& (\mathcal{F}, s) \vDash K_i\psi \\
\iff & \forall t \text{ of } \mathcal{F} \text{ s.t. } s \cap O_i = t \cap O_i : (\mathcal{F}, t) \vDash \psi \qquad \text{by Definition 5} \\
\iff & \forall t \text{ s.t. } t \vDash \theta \text{ and } s \cap O_i = t \cap O_i : (\mathcal{F}, t) \vDash \psi \quad \text{by Definition 4} \\
\iff & \forall t \text{ s.t. } s \cap O_i = t \cap O_i \text{ and } t \vDash \theta : t \vDash \|\psi\|_{\mathcal{F}} \quad \text{by induction hypothesis} \\
\iff & \forall t \text{ s.t. } s \cap O_i = t \cap O_i : t \vDash \theta \to \|\psi\|_{\mathcal{F}} \\
\iff & s \vDash \forall(V \setminus O_i)(\theta \to \|\psi\|_{\mathcal{F}})
\end{array}
$$

For the common knowledge case $\varphi = C_\Delta\psi$, let Λ be the operator defined in as in Definition 6. Also let $\Lambda^0(\alpha) := \alpha$ and $\Lambda^{k+1}(\alpha) := \Lambda(\Lambda^k(\alpha))$.

For left to right, suppose $(\mathcal{F}, s) \vDash C_\Delta\psi$. Note that Λ is monotone increasing but there are only finitely many boolean functions over V. Hence there is some m such that $\mathbf{gfp}\Lambda = \Lambda^m$. Therefore we can show $s \vDash \mathbf{gfp}\Lambda$ by proving $s \vDash \Lambda^m(\top)$ for any m. Suppose not, i.e. there is an m such that $s \nvDash \Lambda^m(\top)$. Then $s \nvDash \|\psi\|_{\mathcal{F}}$ or $s \nvDash \bigwedge_{i\in\Delta} \forall(V \setminus O_i)(\theta \to \Lambda^{m-1}(\top))$. The first is excluded by the induction hypothesis applied to $(\mathcal{F}, s) \vDash \psi$ which follows from $(\mathcal{F}, s) \vDash C_\Delta\psi$. Hence there must be some $i \in \Delta$ and an assignment s_2 such that $s \cap O_i = s_2 \cap O_i$ and $s_2 \nvDash \theta \to \Lambda^{m-1}(\top)$. Then $s_2 \vDash \theta$, so s_2 is a state of $\mathcal{F}$, and $s_2 \nvDash \Lambda^{m-1}(\top)$. Spelling this out we have $s_2 \nvDash \|\psi\|_{\mathcal{F}}$ or $s_2 \nvDash \bigwedge_{i\in\Delta} \forall(V \setminus O_i)(\theta \to \Lambda^{m-2}(\top))$. Again the first case cannot be: s_2 is a state of $\mathcal{F}$ and by $s_1 \cap O_i = s_2 \cap O_i$ we have $(s, s_2) \in \mathcal{E}_\Delta$. Thus $(\mathcal{F}, s) \vDash C_\Delta\psi$ implies $(\mathcal{F}, s_2) \vDash \psi$ which by induction hypothesis gives $s_2 \vDash \|\psi\|_{\mathcal{F}}$. Iterating this we get an $\mathcal{E}_\Delta$-chain $s = s_1, \ldots, s_m$ such that $s_{1+k} \vDash \|\psi\|_{\mathcal{F}}$ and $s_{1+k} \nvDash \Lambda^{m-k}(\top)$ for all $k \in \{1, \ldots, m-1\}$. In particular $s_m \nvDash \Lambda(\top)$ and because $s_m \vDash \|\psi\|_{\mathcal{F}}$ we get $s_k \nvDash \top$. Contradiction! Hence $s \vDash \Lambda^m(\top)$ must hold for all m.

For right to left, suppose $s \vDash \mathbf{gfp}\Lambda$. Note that $\mathbf{gfp}\Lambda \to \Lambda^k(\top)$ is valid and thus we have $s \vDash \Lambda^k(\top)$ for any k. Fix any state t of $\mathcal{F}$ such that $(s, t) \in \mathcal{E}^*_\Delta$. We have to show $(\mathcal{F}, t) \vDash \psi$. By definition of $\mathcal{E}^*_\Delta$ there is a chain $s = s_1, \ldots, s_m = t$ and there are agents $i_1, \ldots, i_{m-1} \in \Delta$ such that for all $k \in \{1, \ldots, m-1\}$ we have $s_k \cap O_{i_k} = s_{k+1} \cap O_{i_k}$. Note $s = s_1$ and $s_1 \vDash \Lambda^m(\top)$, i.e. $s_1 \vDash \|\psi\|_{\mathcal{F}} \wedge \bigwedge_{i\in\Delta} \forall(V \setminus O_i)(\theta \to \Lambda^{m-1}(\top))$. This implies $s_1 \vDash \forall(V \setminus O_{i_1})(\theta \to \Lambda^{m-1}(\top))$. By $s_1 \cap O_{i_1} = s_2 \cap O_{i_1}$ we get $s_2 \vDash \theta \to \Lambda^{m-1}(\top)$. Because s_2 is a state of $\mathcal{F}$ we have $s_2 \vDash \theta$ and therefore $s_2 \vDash \Lambda^{m-1}(\top)$. Iterating this, we get $s_{1+k} \vDash \Lambda^{m-k}(\top)$ for all $k \in \{1, \ldots, m-1\}$. In particular $s_m \vDash \Lambda(\top)$ which implies $s_m \vDash \|\psi\|_{\mathcal{F}}$. By $s_m = t$ and the induction hypothesis this shows $(\mathcal{F}, t) \vDash \psi$.

For public announcements $\varphi = [\psi]\xi$ note the following equivalences:

$$
\begin{array}{ll}
& (\mathcal{F}, s) \vDash [\psi]\xi \\
\iff & (\mathcal{F}, s) \vDash \psi \text{ implies } (\mathcal{F}^\psi, s) \vDash \xi \quad \text{by Definition 5} \\
\iff & s \vDash \|\psi\|_{\mathcal{F}} \text{ implies } s \vDash \|\xi\|_{\mathcal{F}^\psi} \qquad \text{by induction hypothesis} \\
\iff & s \vDash \|\psi\|_{\mathcal{F}} \to \|\xi\|_{\mathcal{F}^\psi}
\end{array}
$$

Similarly, for group announcements:

$$
\begin{aligned}
& (\mathcal{F}, s) \vDash [\psi]_{\Delta} \xi \\
\iff\ & (\mathcal{F}, s) \vDash \psi \text{ implies } (\mathcal{F}^{\Delta}_{\psi}, s \cup \{p_\psi\}) \vDash \xi && \text{by Definition 5} \\
\iff\ & s \vDash \|\psi\|_{\mathcal{F}} \text{ implies } s \cup \{p_\psi\} \vDash \|\xi\|_{\mathcal{F}^{\Delta}_{\psi}} && \text{by induction hypothesis} \\
\iff\ & s \vDash \|\psi\|_{\mathcal{F}} \text{ implies } s \vDash (\|\xi\|_{\mathcal{F}^{\Delta}_{\psi}})(\tfrac{p_\psi}{\top}) \\
\iff\ & s \vDash \|\psi\|_{\mathcal{F}} \to (\|\xi\|_{\mathcal{F}^{\Delta}_{\psi}})(\tfrac{p_\psi}{\top})
\end{aligned}
$$

□

We can now also explain the semantics for public and group announcements given in Definition 5. First observe that *public* announcements only modify the state law of the knowledge structure. Moreover, the new state law is always a conjunction containing the previous one. Hence the set of states is restricted, just like public announcements on Kripke models restrict the set of possible worlds. Second, note that a group announcement adds a single observational variable which is then linked to a formula evaluated on the previous structure. Hence the number of states is constant, as in the Kripke semantics in Definition 3.

Both announcements use the local boolean equivalent of the announced formula with respect to the original structure $\mathcal{F}$, just like in Kripke semantics the condition for copying worlds or cutting edges is about the original $\mathcal{M}$ and not the model after the announcement. A well-known consequence of this hence also holds about our knowledge structures: Truthful announcements can be unsuccessful in the sense that after something is announced it is not true anymore. Famous examples are Moore sentences of the form "It snows and you don't know it".

Theorem 1 is somewhat surprising as it "explains away" dynamic and epistemic operators. But it does not make DEL any less expressive. Rather we can think of the original formulas as universally usable — they capture an intended meaning across different models or structures. Their local boolean equivalents given by Definition 6 still do so across states, but only within a specific structure.

4 Example: Muddy Children

How does our new format do in practice? For this purpose, we consider some well-known benchmarks in the epistemic agency literature. We start with how their new representations looks like. After that, we go on to actual computational experiments. The famous Muddy Children example will illustrate how announcements, both of propositional and of epistemic facts, work on knowledge structures. An early version of this puzzle are the three ladies on a train in [30]. For a standard analysis with Kripke models, see [24, p. 24-30] or [14, p. 93-96].

Let p_i stand for "child i is muddy". We consider the case of three children $I = \{1, 2, 3\}$ who are all muddy, i.e. the actual state is $\{p_1, p_2, p_3\}$. At the beginning the children do not have any information, hence the initial knowledge structure $\mathcal{F}_0$ in Figure 2 has the state law $\theta_0 = \top$. The set of states of $\mathcal{F}_0$ is therefore the full powerset of the vocabulary, i.e. $\mathcal{P}(\{p_1, p_2, p_3\})$. All children can observe whether the others are muddy but do not see their own face. This is

represented with observational variables: Agent 1 observes p_2 and p_3, etc. Now the father says: "At least one of you is muddy." This public announcement limits the set of states by adding this statement to the state law. Note that it already is a purely boolean statement, hence the formula is added as it is, leading to the new knowledge structure $\mathcal{F}_1$ as shown in Figure 2.

$$\mathcal{F}_0 = \left(\begin{array}{r} O_1 = \{p_2, p_3\} \\ V = \{p_1, p_2, p_3\}, \theta_0 = \top, \; O_2 = \{p_1, p_3\} \\ O_3 = \{p_1, p_2\} \end{array} \right)$$

$$\mathcal{F}_1 = \left(\begin{array}{r} O_1 = \{p_2, p_3\} \\ V = \{p_1, p_2, p_3\}, \theta_1 = (p_1 \vee p_2 \vee p_3), \; O_2 = \{p_1, p_3\} \\ O_3 = \{p_1, p_2\} \end{array} \right)$$

Fig. 2. Knowledge structures before and after the first announcement.

The father now asks "Do you know if you are muddy?" but none of the children does. As it is common in the literature, we understand this as a public announcement of "Nobody knows their own state.": $\bigwedge_{i \in I}(\neg(K_i p_i \vee K_i \neg p_i))$. This is not a purely boolean formula, hence the public announcement is slightly more complicated: Using Definition 6 and Theorem 1 we find a boolean formula which on the current knowledge structure $\mathcal{F}_1$ is equivalent to the announced formula. Then this boolean equivalent is added to θ. We have

$$\begin{aligned} \|K_1 p_1\|_{\mathcal{F}_1} &= \forall(V \setminus O_1)(\theta_1 \to \|p_1\|_{\mathcal{F}_1}) \\ &= \forall p_1((p_1 \vee p_2 \vee p_3) \to p_1) \\ &= ((\top \vee p_2 \vee p_3) \to \top) \wedge ((\bot \vee p_2 \vee p_3) \to \bot) \\ &= \neg(p_2 \vee p_3) \end{aligned}$$

$$\begin{aligned} \|K_1 \neg p_1\|_{\mathcal{F}_1} &= \forall(V \setminus O_1)(\theta_1 \to \|\neg p_1\|_{\mathcal{F}_1}) \\ &= \forall p_1((p_1 \vee p_2 \vee p_3) \to \neg p_1) \\ &= ((\top \vee p_2 \vee p_3) \to \neg\top) \wedge ((\bot \vee p_2 \vee p_3) \to \neg\bot) \\ &= \bot \end{aligned}$$

and analogous for $K_2 p_2$, $K_2 \neg p_2$, $K_3 p_3$ and $K_3 \neg p_3$. These results make intuitive sense: In our situation where all children are muddy, a child knows it is muddy iff it sees that the other two children are clean. It can never know that it is clean itself. The announced formula becomes

$$\begin{aligned} \| \bigwedge_{i \in I} (\neg(K_i p_i \vee K_i \neg p_i)) \|_{\mathcal{F}_1} &= \bigwedge_{i \in I} \|\neg(K_i p_i \vee K_i \neg p_i)\|_{\mathcal{F}_1} \\ &= \neg(\neg(p_2 \vee p_3)) \wedge \neg(\neg(p_1 \vee p_3)) \wedge \neg(\neg(p_1 \vee p_2)) \\ &= (p_2 \vee p_3) \wedge (p_1 \vee p_3) \wedge (p_1 \vee p_2) \end{aligned}$$

The announcement essentially says that at least two children are muddy. We get a knowledge structure $\mathcal{F}_2$ with the following more restrictive state law θ_2. Vocabulary and observational variables do not change, so we do not repeat them.

$$\theta_2 = (p_1 \vee p_2 \vee p_3) \wedge ((p_2 \vee p_3) \wedge (p_1 \vee p_3) \wedge (p_1 \vee p_2))$$

Now the same announcement ("Nobody knows their own state.") is made again. It is important that again we start with the epistemic formula $\bigwedge_{i\in I}(\neg(K_i p_i \vee K_i \neg p_i))$ and compute an equivalent formula with respect to $\mathcal{F}_2$. Now by further boolean reasoning we have that

$$\begin{aligned}
|K_1 p_1|_{\mathcal{F}_2} &= \forall(V \setminus O_1)(\theta_2 \to |p_1|_{\mathcal{F}_2}) \\
&= \forall p_1((p_1 \vee p_2 \vee p_3) \wedge ((p_2 \vee p_3) \wedge (p_1 \vee p_3) \wedge (p_1 \vee p_2)) \to p_1) \\
&= ((\top \vee p_2 \vee p_3) \wedge ((p_2 \vee p_3) \wedge (\top \vee p_3) \wedge (\top \vee p_2)) \to \top) \\
&\quad \wedge((\bot \vee p_2 \vee p_3) \wedge ((p_2 \vee p_3) \wedge (\bot \vee p_3) \wedge (\bot \vee p_2)) \to \bot) \\
&= \top \wedge ((p_2 \vee p_3) \wedge ((p_2 \vee p_3) \wedge p_3 \wedge p_2) \to \bot) \\
&= \neg((p_2 \vee p_3) \wedge ((p_2 \vee p_3) \wedge p_3 \wedge p_2)) \\
&= \neg(p_3 \wedge p_2)
\end{aligned}$$

$$\begin{aligned}
|K_1 \neg p_1|_{\mathcal{F}_2} &= \forall(V \setminus O_1)(\theta_2 \to |\neg p_1|_{\mathcal{F}_2}) \\
&= \forall p_1(\theta_2 \to \neg p_1) \\
&= \forall p_1((p_1 \vee p_2 \vee p_3) \wedge ((p_2 \vee p_3) \wedge (p_1 \vee p_3) \wedge (p_1 \vee p_2)) \to \neg p_1) \\
&= ((\top \vee p_2 \vee p_3) \wedge ((p_2 \vee p_3) \wedge (\top \vee p_3) \wedge (\top \vee p_2)) \to \neg\top) \\
&\quad \wedge((\bot \vee p_2 \vee p_3) \wedge ((p_2 \vee p_3) \wedge (\bot \vee p_3) \wedge (\bot \vee p_2)) \to \neg\bot) \\
&= (\top \wedge ((p_2 \vee p_3) \wedge \top \wedge \top) \to \bot) \\
&\quad \wedge((p_2 \vee p_3) \wedge ((p_2 \vee p_3) \wedge (p_3) \wedge (p_2)) \to \top) \\
&= (p_2 \vee p_3 \to \bot) \wedge \top \\
&= \neg(p_2 \vee p_3)
\end{aligned}$$

which together gives us

$$\begin{aligned}
\|\neg(K_1 p_1 \vee K_1 \neg p_1)\|_{\mathcal{F}_2} &= \neg(\|K_1 p_1\|_{\mathcal{F}_2} \vee \|K_1 \neg p_1\|_{\mathcal{F}_2}) \\
&= \neg(\neg(p_3 \wedge p_2) \vee \neg(p_2 \vee p_3)) \\
&= (p_3 \wedge p_2) \wedge (p_2 \vee p_3) \\
&= p_3 \wedge p_2
\end{aligned}$$

and analogous formulas for children 2 and 3. Note that this admittedly tedious calculation brings to light a detail of the puzzle: It would suffice to announce "I do not know *that* I am muddy", in contrast to "I do not know *whether* I am muddy" which in general is more informative but not in this specific situation.

Finally, with respect to $\mathcal{F}_2$ we get the following boolean equivalent of the announcement, essentially saying that everyone is muddy.

$$\begin{aligned}
\| \bigwedge_{i\in I}(\neg(K_i p_i \vee K_i \neg p_i))\|_{\mathcal{F}_2} &= (p_3 \wedge p_2) \wedge (p_3 \wedge p_1) \wedge (p_2 \wedge p_1) \\
&= p_1 \wedge p_2 \wedge p_3
\end{aligned}$$

The resulting knowledge structure thus has the state law $\theta_3 = \theta_2 \wedge (p_1 \wedge p_2 \wedge p_3)$ which is in fact equivalent to $p_1 \wedge p_2 \wedge p_3$ and marks the end of the story: The only state left is the situation in which all three children are muddy. Moreover, this is common knowledge among them because the only state is also the only state reachable via $\mathcal{E}_I^*$ in Definition 5. Alternatively, note that the fixed point mentioned in Definition 6 in this case will be the same as θ_3.

5 Implementation and Benchmarks

The previous section showed how epistemic operators get replaced by booleans when a new state law is computed. We could see that syntactically the state law becomes more and more complex, but semantically the same boolean function can be represented with a much shorter formula. This is where Binary Decision Diagrams (BDDs) come in extremely handy. They were first presented in [7] and provide an elegant data structure for boolean functions.

Definition 7. *A binary decision diagram for a vocabulary V is a directed acyclic graph where non-terminal nodes are from V with two outgoing edges and terminal nodes are $\top$ or $\bot$. A binary decision diagram is* ordered *according to a total order $<$ of V if for any edge from a node p to a node q we have $p \leq q$. A binary decision diagram is* reduced *iff it does not contain two isomorphic subgraphs. By the abbreviation* BDD *we always mean an ordered and reduced binary decision diagram. A total order on the propositional variables uniquely determines the BDD of a boolean formula, and we will use $\mathsf{Bdd}(\varphi)$ for the BDD of φ.*

Example 5. Consider the boolean function given by the formula $\neg(p_1 \wedge \neg p_2) \to p_3$. Figure 3 shows a full decision tree for this function and the BDD obtained by identifying all isomorphic subgraphs.

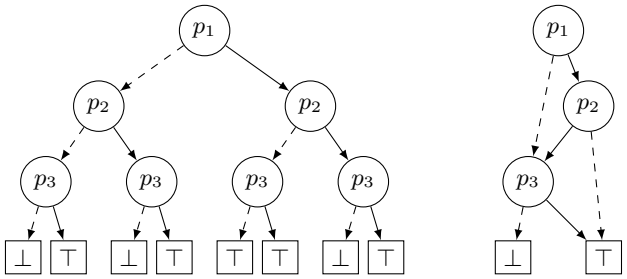


Fig. 3. Full decision tree and BDD of $\neg(p_1 \wedge \neg p_2) \to p_3$

Note that we do not lose any information. To check whether a given assignment satisfies the function represented by this BDD, we start at the root and follow the outgoing arrows as follows: If the variable at the current node is true according to the given assignment, follow the solid arrow, otherwise the dashed one. For example we can check that $\{p_1, p_3\} \vDash \neg(p_1 \wedge \neg p_2) \to p_3$ by following the right arrow twice. Note that the BDD then does not even ask about the value of p_3. This reflects that we also have $\{p_1\} \vDash \neg(p_1 \wedge \neg p_2) \to p_3$.

BDDs have several advantages. In many cases they are less redundant and thus smaller than a corresponding truth table. Additionally, they can be manipulated efficiently: Given BDDs of φ and ψ we can compute the BDD of $\varphi \wedge \psi$, $\varphi \to \psi$

etc. Moreover, BDDs are canonical: Two formulas are equivalent iff their BDDs are identical. In particular, once we have the BDD of a formula it is easy to check whether it is a tautology or a contradiction: A formula is constant iff its BDD consists of a single terminal node $\top$ or $\bot$. For an in-depth introduction, see [29, p. 202-280]. To see how BDDs can be used to describe knowledge structures, Figure 4 shows the BDDs of the state laws θ_0 to θ_3 from the Muddy Children example in Section 4.

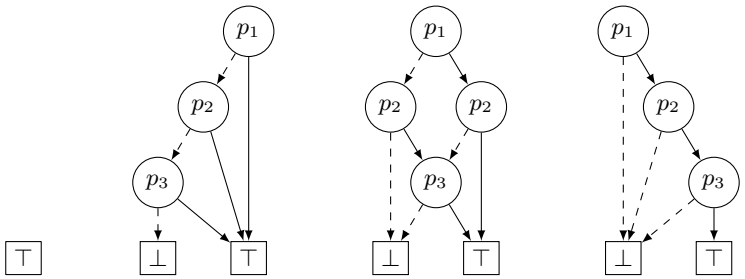


Fig. 4. Four BDDs representing the state laws θ_0 to θ_3.

Our new symbolic model checker SMCDEL works as follows: It takes two inputs, a scene $(\mathcal{F}, s)$ where the state law is given as a BDD, and a DEL formula φ. To check whether φ holds at state s we first compute the equivalent boolean formula $\|\varphi\|_{\mathcal{F}}$ according to Definition 6 and then check the boolean satisfaction $s \vDash \|\varphi\|_{\mathcal{F}}$. Alternatively, we can check whether a formula is valid on $\mathcal{F}$, i.e. true at *all* states, by checking whether $\theta \to \|\varphi\|_{\mathcal{F}}$ is a tautology. The full set of states does not have to be generated and events are not executed explicitly.

Alternatively, the input can be a Kripke model and a formula to be checked. SMCDEL will then first build an equivalent knowledge structure using the construction given in Definition 9 below.

The model checker is implemented in Haskell and can be used similarly to DEMO-S5 both in the interactive compiler `ghci` and compiled as a library. Additionally we provide a simple text interface which is described in the Appendix. To represent BDDs we use CacBDD [34] via the binding library HasCacBDD [26]. The program can also be used with CUDD [37] which provides very similar performance. All of the following experiments and benchmarks were done using 64-bit Debian GNU/Linux 8 with kernel 3.16.0–4, GHC 7.10.3 and g++ 4.9 on an Intel Core i3–2120 3.30 GHz processor and 4 GB of memory. To get precise timing results we used the Haskell library Criterion [36].

Muddy Children

We compared the performance of this method to DEMO-S5, an explicit model checker optimized for multi-agent S5 [19]. As a benchmark we used the question

"For n muddy children, how many announcements of »Nobody knows their own state.« are needed until they do know their own state?". We measured how long each method takes to find and verify the correct answer $(n-1)$ by iteratively evaluating DEL formulas saying that after this many announcements nobody/everybody knows their own state. More details about the exact input can be found in the appendix and the technical report accompanying SMCDEL.

Figure 5 shows the results on a logarithmic scale: Explicit model checking with DEMO-S5 quickly becomes unfeasible whereas our symbolic model checker SMCDEL deals with scenarios up to 40 agents in less than a second.

Note that this is a comparison of software and representation at the same time: While DEMO-S5 uses a Kripke model, SMCDEL uses the knowledge structure which we discussed in the previous section. The speedup could therefore arise at different steps: First an initial knowledge structure or Kripke model is generated, then it will be updated because the formula to be evaluated starts with an announcement. Finally, a formula is evaluated on the result.

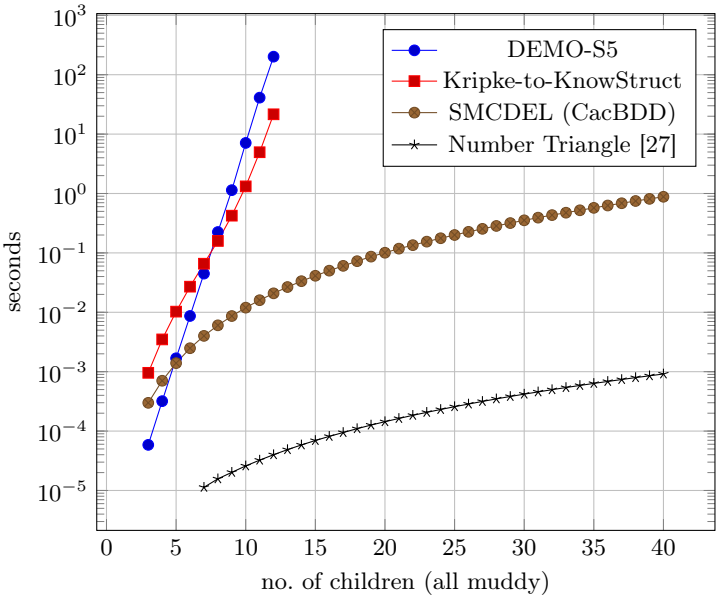


Fig. 5. Benchmark Results on a logarithmic scale.

To test in which of the steps our new implementation is faster we also benchmarked a variant of SMCDEL which takes a Kripke model as input. It uses the translation from Definition 6 to construct an equivalent knowledge structure and checks the given formula on that structure. The results are "Kripke-to-KnowStruct" in Figure 5. We can see that the performance of this method is worse than DEMO-S5 for small instances but becomes slightly better for nine or

more agents. This reveals that the standard semantics are slow because generation of large Kripke models takes a long time and not the evaluation of updates and formulas afterwards.

In some sense this is where theory and practice of model checking part ways, because only the evaluation of formulas is considered part of "model checking" itself, not the time to generate or read in the description of the model. In particular, the computational complexity of model checking is measured with the size of the model as a parameter [1]. But this size will depend heavily on the representation: The Kripke model for situations like the Muddy Children grows exponentially in the number of agents, so even if model checking takes time polynomial in the size of the model, it is exponential in the number of agents. In contrast, consider the size of a knowledge structure: For n Muddy Children the initial model is given by $(\{p_1, \ldots, p_n\}, \top, O_1 = \{p_1\}, \ldots, O_n = \{p_n\})$ which we can write as a string of length $\mathcal{O}(n^2)$. Moreover, the BDDs describing intermediate state laws will maximally have $\mathsf{ceiling}(\frac{n}{2})^2$ many nodes.

We also implemented and benchmarked an alternative modeling of Muddy Children given in [27]. Inspired by the number triangle, the authors use models without names or indices for agents. Only two kinds of agents, the muddy and non-muddy children, are distinguished. Moreover, instead of epistemic relations the model contains observational-states which describe the perspective of a type of agents. This yields a model for n agents with only $2n + 1$ instead of 2^n states as shown in Figure 6 for the case $n = 3$.

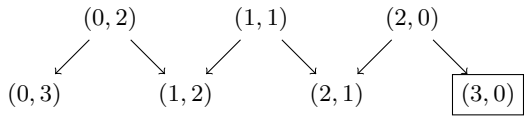


Fig. 6. Triangle model for Muddy Children.

However, [27] does not provide formal syntax and semantics and it is not possible to evaluate standard DEL on triangle models. For our implementation we chose the language $\varphi ::= \neg\varphi \mid \varphi \wedge \varphi \mid Q \mid K_b \mid \overline{K}_b$ where Q is a generalized quantifier, b is a bit for muddy or non-muddy, K_b means that all agents of kind b know their own state and $\overline{K}_b$ means that all agents of kind b do *not* know their own state. Note that the knowledge operators do not take any formula arguments and their semantics start with a universal quantifier. This also explains why $\overline{K}_b$ is needed to express the announcement "Nobody knows their own state". In contrast, $\neg K_b$ would mean that there is at least one agent not knowing their own state. Also note that updates need to be interpreted differently: The first announcement "At least one of you is muddy." is a quantifier and removes the state $(0, 3)$ in Figure 6. After that the announcements of "Nobody knows their own state." are given by $\overline{K}_0 \wedge \overline{K}_1$ and will remove observational states in the upper layer. For more details, see the SMCDEL documentation.

The performance of this number triangle model is impressive. However, the modeling is very specific to the Muddy Children, while DEMO-S5 and SMCDEL are general DEL model checkers. Similar abstractions and concise models might also be found for other examples discussed in this section, but they need to be constructed for each specific case. Still, we see the results as a motivation to study abstraction methods such as agent kinds in the future.

Muddy Children has also been used to benchmark MCMAS [31] but the formula checked there concerns the correctness of behavior and not how many rounds are needed. Moreover, the interpreted system semantics of model checkers like MCMAS are very different from DEL. Still, connections between DEL and temporal logics have been studied and translations are available [5,16]. The next example is more suited for a direct comparison with MCMAS.

The Dining Cryptographers

A scenario which fits nicely into both frameworks is the dining cryptographers protocol described in [9]: Three cryptographers go out to have diner. After a delicious meal the waiter tells them that the bill has already been paid. The cryptographers know that either it was one of them or the NSA. They want to find which of the two is the case but also respect the wish to stay anonymous: if one of them paid they do not want that person to be revealed.

To accomplish this, they can use the following protocol: For every pair of cryptographers a coin is flipped in such a way (e.g. under the table) that only those two see the result. Then they announce whether the two coins they saw were different or the same. But, there is an exception: If one of them paid, then this person says the opposite. After these announcements are made, the cryptographers can infer that the NSA paid iff the number of people saying that they saw the same result on both coins is even. More formally, they use boolean variables and the XOR function. For details how to model this protocol using DEL, see [21].

The statement "If cryptographer 1 did not pay the bill, then after the announcements are made, he knows that no cryptographers paid, or that someone paid, but in this case he does not know who did." is also checked in [31]. Following the translation ideas in [5,16] we can formalize the same statement as

$$\neg p_1 \to [\psi](K_1(\bigwedge_{i=1}^{n} \neg p_i) \vee (K_1(\bigvee_{i=2}^{n} p_i) \wedge \bigwedge_{i=2}^{n} (\neg K_1 p_i)))$$

where p_i says that agent i paid and ψ is the announcement.

Table 1 shows how many propositions we need to model the situation for n agents and how long SMCDEL needs to run to check the above statement. In particular this is faster than the equivalent computations using temporal logics and the model checker MCMAS which takes about 5 seconds for 40 agents [31, Tables 4]. Figure 17 in the appendix lists the input for SMCDEL for the case of three agents.

n	Propositions	Seconds
10	56	0.0017
20	211	0.0092
40	821	0.0739
80	3241	0.9751
120	7261	3.2806
160	12881	8.1046

Table 1. Results for n Dining Cryptographers.

Russian Cards

As another case study we applied our symbolic model checker to the Russian Cards Problem. One of its first logical analyses is [13] and the problem has since gained notable attention as an intuitive example of information-theoretically (in contrast to computationally) secure cryptography [12,17].

The basic version of the problem is this: Seven cards, enumerated from 0 to 6, are distributed between Anne, Bob and Crow such that Anne and Bob both receive three cards and Crow one card. It is common knowledge which cards exist and how many cards each agent has. Everyone knows their own but not the others' cards. The goal of Anne and Bob now is to learn each others cards without Crow learning them. They can only communicate via public announcements.

Many different solutions exist but here we will focus on the so-called five-hands protocols (and their extensions with six or seven hands): First Anne makes an announcement of the form "My hand is one of these: ...". If her hand is 012 she could for example take the set $\{012, 034, 056, 135, 146, 236\}$. It can be checked that this announcement does not tell Crow anything, independent of which card it has. In contrast, Bob will be able to rule out all but one of the hands in the list depending on his own hand. Hence the second and last step of the protocol is an announcement by Bob about which card Crow has. For example, if Bob's hand is 345 he would finish the protocol with "Crow has card 6.".

Verifying this protocol for the fixed deal 012|345|6 with our symbolic model checker takes less than a second. Compared to that, a DEMO implementation [15] needs 4 seconds to check one protocol.

Moreover, checking multiple protocols in a row does not take much longer because the BDD package caches results. We can not just verify but also find all 5/6/7-hands protocols, using the following combination of manual reasoning and brute-force. By Proposition 32 in [13] safe announcements from Anne never contain "crossing" hands, i.e. two hands with multiple card in common. If we also assume that the hands are lexicographically ordered, this leaves us with 1290 possible lists of five, six or seven hands of three cards. Only some of them are safe announcements which can be used by Anne. We can find them by checking all the corresponding 1290 formulas. Our model checker can filter out the 102 safe announcements within 1.6 seconds, generating and verifying the same list as in [13] where it was fully manually generated.

Going one step further, suppose we do not know anything like Proposition 32 from [13] that allows us to restrict the search space. This perspective was also adopted in [23], turning the puzzle into an epistemic planning problem. If we only fix that Alice will announce five hands, including her own which w.l.o.g. is 012, then she has to pick four other hands of three cards each. The number of possible actions is then 46376. It takes our model checker about 160 seconds to find the 60 safe announcements among them. Finally, if we also relax the condition that Bob will answer with "Crow has card 6" but instead consider "Crow has card n" for any card n, the search space grows by a factor of 7 to 324632. It now takes around 20 minutes to find the solutions. Note however, that none of the additional plans are successful, hence the same 60 plans are generated.

Sum and Product

Maybe the most famous example in the DEL literature after the Muddy Children is the Sum and Product puzzle [25, translated from Dutch]:

> A says to S and P: "I chose two numbers x, y such that $1 < x < y$ and $x + y \leq 100$. I will tell $s = x + y$ to S alone, and $p = xy$ to P alone. These messages will stay secret. But you should try to calculate the pair (x, y)". He does as announced. Now follows this conversation: P says: "I do not know it." S says: "I knew that." P says: "Now I know it." S says: "No I also know it." Determine the pair (x, y).

Our model checker can also solve this classic and we can improve upon the results of existing implementations. However, this comes with a trade-off in convenience: In DEMO-S5 [19] Kripke models are parameterized with a type. This allows the user to encode information in the states directly. For example, the states in a model for Sum and Product can be pairs of integers.

In contrast, because of the underlying BDD representation our model has to be completely propositional. In the future we plan to implement typed Kripke models using similar encodings like the following manual translation.

To represent numbers we use binary encodings for x, y, $s := x+y$ and $p := x*y$. Let $\mathsf{ceiling}(\cdot)$ denote the smallest natural number not less than the argument. We need $\mathsf{ceiling}(\log_2 N)$ propositions for every variable that should take values up to N. For example, to represent the possible values of $x \in \{2, \ldots, 100\}$ we can use $p_1, \ldots, p_7$. The statement $p = 5$ is then encoded as $p_1 \wedge p_2 \wedge p_3 \wedge p_4 \wedge \neg p_5 \wedge p_6 \wedge \neg p_7$. Given this encoding we have propositional formulas for $x = n$ etc. and can use them to formalize the puzzle as usual [14, Section 4.11]. The state law for Sum and Product is a big disjunction over all possible pairs of x and y with the given restrictions. It is here where we ensure that s and p are actually the sum and the product of n and m:

$$\bigvee \{x = n \wedge y = m \wedge s = n + m \wedge p = n \cdot m \mid 2 \leq n < m \leq 100, n + m \leq 100\}$$

To let the agents S and P know the values of s and p respectively, we define the observational variables $O_S := \{s_1, \ldots, s_7\}$ and $O_P := \{p_1, \ldots, p_7\}$. Now we

can use the usual formulas to say that an agent knows a variable and that the statements of the dialogue can be truthfully announced. Our model checker can then be asked in which states the DEL formula characterizing a solution holds. It takes less than two seconds to find the unique solution. In particular this is faster than a previous implementation in [33] which is also based on BDDs. However, it is still slower than an optimized version of explicit model checking with DEMO-S5 which can do it in less than one second. This is probably due to a well-known problem already mentioned in [7]: BDD representations of products tend to be larger. Our program thus spends most of its time to build the BDD of the state law including the restriction $p = n \cdot m$ before it can actually check any given formula.

6 Equivalence of the two Semantics

Having shown the computational advantage of our new modeling, we now look more deeply into the foundations of what we have been doing. For a start, we show that knowledge structures and standard models for DEL are equivalent from a semantic point of view. Lemma 1 gives us a canonical way to show that a knowledge structure and an S5 Kripke model satisfy the same formulas. Theorems 2 and 3 say that such equivalent models and structures can always be found. These translations are also implemented in SMCDEL.

Lemma 1. *Suppose we have a knowledge structure $\mathcal{F} = (V, \theta, O_1, \ldots, O_n)$ and a finite S5 Kripke model $M = (W, \pi, \mathcal{K}_1, \ldots, \mathcal{K}_n)$ with a set of primitive propositions $U \subseteq V$. Furthermore, suppose we have a function $g : W \to \mathcal{P}(V)$ such that*

C1 For all $w_1, w_2 \in W$, and all i such that $1 \leq i \leq n$, we have that $g(w_1) \cap O_i = g(w_2) \cap O_i$ iff $w_1 \mathcal{K}_i w_2$.
C2 For all $w \in W$ and $p \in U$, we have that $p \in g(w)$ iff $\pi(w)(p) = \top$.
C3 For every $s \subseteq V$, s is a state of $\mathcal{F}$ iff $s = g(w)$ for some $w \in W$.

Then, for every $\mathcal{L}(U)$-formula φ we have $(\mathcal{F}, g(w)) \vDash \varphi$ iff $(\mathcal{M}, w) \vDash \varphi$.

Before we dive into the proof, let us step back a bit to see that conditions C1 to C3 describe a special case of something well-known, namely a p-morphism between the model $\mathcal{M}$ and a model encoded by the structure $\mathcal{F}$ which we will make precise in Definition 8 below. The mathematically reader might thus already be convinced by general invariance results [6, §2.1] and skip the following induction.

Proof. We proceed by induction on φ. First consider the base case when φ is a primitive proposition, say p. Then, by condition C2, we have that $(\mathcal{F}, g(w)) \vDash p$ iff $p \in g(w)$ iff $\pi(w)(p) = \top$ iff $(\mathcal{M}, w) \vDash p$.

Now suppose that φ is not a primitive proposition and as an induction hypothesis the claim holds for every formula of lower complexity than φ. We distinguish five cases:

1. φ is of form $\neg\psi$ or $\psi \wedge \xi$. Definitions 3 and 5 do the same recursion for negations and conjunctions, hence this follows by the induction hypothesis.

2. φ is of form $K_i\psi$. By Definition 5, we have $(\mathcal{F}, g(w)) \vDash K_i\psi$ iff $(\mathcal{F}, s) \vDash \psi$ for all states s of $\mathcal{F}$ with $g(w) \cap O_i = s \cap O_i$. By C3 this is equivalent to having $(\mathcal{F}, g(w')) \vDash \psi$ for all $w' \in W$ with $g(w) \cap O_i = g(w') \cap O_i$, which by C1 is equivalent to $(\mathcal{F}, g(w')) \vDash \psi$ for all $w' \in W$ with $w\mathcal{K}_i w'$. Now by the induction hypothesis, this is equivalent to $(\mathcal{M}, w') \vDash \psi$ for all $w' \in W$ with $w\mathcal{K}_i w'$ which is exactly $(\mathcal{M}, w) \vDash K_i\psi$ by Definition 3.
3. φ is of form $C_\Delta\psi$. Recall that, for arbitrary states s and t of $\mathcal{F}$, $(s,t) \in \mathcal{E}_\Delta$ iff there exists an $i \in \Delta$ with $s \cap O_i = t \cap O_i$. By C1, for all $w_1, w_2 \in W$,

$$(g(w_1), g(w_2)) \in \mathcal{E}_\Delta \text{ iff } (w_1, w_2) \in \bigcup_{i \in \Delta} R_i.$$

   As $\mathcal{E}^*_{\mathcal{V}_\Delta}$ is the transitive closure of $\mathcal{E}_{\mathcal{V}_\Delta}$, and $\mathcal{C}^M_\Delta$ is that of $\bigcup_{i\in\Delta} R_i$, by C3 we have for all $w_1, w_2 \in W$ that

$$(g(w_1), g(w_2)) \in \mathcal{E}^*_\Delta \text{ iff } (w_1, w_2) \in \mathcal{C}^M_\Delta$$

   We now claim that $(\mathcal{F}, g(w)) \vDash C_\Delta\psi$ iff $(\mathcal{M}, w) \vDash C_\Delta\psi$. On one hand, $(\mathcal{F}_M, g(w)) \vDash C_\Delta\psi$ iff for all states s of $\mathcal{F}_M$ with $(g(w), s) \in \mathcal{E}^*_\Delta$, $(\mathcal{F}_M, s) \vDash \psi$. By C3, we have that $(\mathcal{F}_M, g(w)) \vDash C_\Delta\psi$ iff for all $w' \in W$ with $(g(w), g(w')) \in \mathcal{E}^*_\Delta$. On the other hand, $(\mathcal{M}, w) \vDash C_\Delta\psi$ iff for all $w' \in W$ with $(w, w') \in \mathcal{C}^M_\Delta$. Hence the claim follows by the above discussion and the induction hypothesis.
4. φ is of form $[\psi]\xi$. By Definition 3, we have that $(\mathcal{M}, w) \vDash [\psi]\xi$ iff $(\mathcal{M}, w) \vDash \psi$ implies $(\mathcal{M}^\psi, w) \vDash \xi$, and by Definition 5 we have that $(\mathcal{F}, g(w)) \vDash [\psi]\xi$, iff $(\mathcal{F}, g(w)) \vDash \psi$ implies $(\mathcal{F}^\psi, g(w)) \vDash \xi$. As $(\mathcal{M}, w) \vDash \psi$ iff $(\mathcal{F}, g(w)) \vDash \psi$ by the induction hypothesis, it suffices to prove that $(\mathcal{M}^\psi, w) \vDash \xi$ iff $(\mathcal{F}^\psi, g(w)) \vDash \xi$. Let g' be the restriction of g to $W^{\mathcal{M}^\psi} = \{w \in W \mid (\mathcal{M}, w) \vDash \psi\}$. Note that because g fulfills the universal conditions C1 and C2, they must also hold for g' with respect to the restricted set $W^{\mathcal{M}^\psi}$. To show C3 for g', for left to right suppose $s \subseteq V$ is a state of $\mathcal{F}^\psi$. Then s is also a state of $\mathcal{F}$ and by condition C3 for g, there is a $w \in W$ such that $s = g(w)$. Moreover, $\mathcal{F}, s \vDash \psi$ and therefore by the induction hypothesis $(\mathcal{M}, w) \vDash \psi$. Hence $w \in W^{\mathcal{M}^\psi}$ and we also have $g'(w) = s$. For right to left suppose $g'(w) = s$ for some $w \in W^{\mathcal{M}^\psi}$ and some $s \subseteq V$. Then $(\mathcal{M}, w) \vDash \psi$ and s is a state of $\mathcal{F}$ because $g(w) = g'(w) = s$. Therefore by the induction hypothesis $\mathcal{F}, s \vDash \psi$. Hence $s \vDash \|\psi\|_{\mathcal{F}}$ which implies that s is also a state of F^ψ. Together, g' fulfills all three conditions and by the induction hypothesis we get that $(\mathcal{M}^\psi, w) \vDash \xi$ iff $(\mathcal{F}^\psi, g(w)) \vDash \xi$.
5. φ is of form $[\psi]_\Delta\xi$. By Definition 3 and 5, we have that $(\mathcal{M}, w) \vDash [\psi]_\Delta\xi$ iff $(\mathcal{M}, w) \vDash \psi$ implies $(\mathcal{M}^\Delta_\psi, w) \vDash \xi$, and $(\mathcal{F}, g(w)) \vDash [\psi]_\Delta\xi$ iff $(\mathcal{F}, g(w)) \vDash \psi$ implies $(\mathcal{F}^\Delta_\psi, \{p_\psi\} \cup g(w)) \vDash \xi$. As $(\mathcal{M}, w) \vDash \psi$ iff $(\mathcal{F}, g(w)) \vDash \psi$ by induction hypothesis, it suffices to prove that $(\mathcal{M}^\Delta_\psi, w) \vDash \xi$ iff $(\mathcal{F}^\Delta_\psi, \{p_\psi\} \cup g(w)) \vDash \xi$. Let $g' : W^{\mathcal{M}^\Delta_\psi} \to \mathcal{P}(V \cup \{p_\psi\})$ be defined by $g'(w) := \{p_\psi\} \cup g(w)$ if $(\mathcal{M}, w) \vDash \psi$ and $g'(w) := g(w)$ otherwise. It remains to show that g' also fulfills C1 to C3. For C1, take any $w_1, w_2 \in W^{\mathcal{M}^\Delta_\psi} = W^M$ and any i. As in Definitions 3 and 5 let $\mathcal{K}_i^{\mathcal{M}^\Delta_\psi}$ and O_i^* be the epistemic relations and observational variables after

the announcement and note the following equivalences where step $\heartsuit$ follows from C1 with respect to g:

$$\begin{aligned}
& g'(w_1) \cap O_i^* = g'(w_2) \cap O_i^* \\
\iff\ & (g(w_1) \cup \{p_\psi \mid M, w_1 \vDash \psi\}) \cap (O_i \cup \{p_\psi \mid i \in \Delta\}) \\
& = (g(w_2) \cup \{p_\psi \mid M, w_2 \vDash \psi\}) \cap (O_i \cup \{p_\psi \mid i \in \Delta\}) \\
\iff\ & g(w_1) \cap O_i = g(w_2) \cap O_i \text{ and } i \in \Delta \Rightarrow (M, w_1 \vDash \psi \text{ iff } M, w_2 \vDash \psi) \\
\overset{\heartsuit}{\iff}\ & w_1 \mathcal{K}_i^M w_2 \text{ and } i \in \Delta \Rightarrow (M, w_1 \vDash \psi \text{ iff } M, w_2 \vDash \psi) \\
\iff\ & w_1 \mathcal{K}_i^{\mathcal{M}_\psi^\Delta} w_2
\end{aligned}$$

To show C2, note that W, U and π do not change in the group announcement part of Definition 3. Thus C2 for g' follows from C2 for g.
For C3, take any $s \subseteq V \cup \{p_\psi\}$. To show left to right, suppose s is a state of $\mathcal{F}_\psi^\Delta$. Then $s \vDash \theta \wedge (p_\psi \leftrightarrow \|\psi\|_\mathcal{F})$. In particular $s \vDash \theta$, so $t := s \cap V$ is a state of $\mathcal{F}$. Hence by C3 for g there is a w such that $g(w) = t$. We consider two cases. First, suppose $p_\psi \in s$. Then by $s \vDash (p_\psi \leftrightarrow \|\psi\|_\mathcal{F})$ we have $s \vDash \|\psi\|_\mathcal{F}$. Note that p_ψ does not occur in $\|\psi\|_\mathcal{F}$, hence $t \vDash \|\psi\|_\mathcal{F}$. By Theorem 1 we have $\mathcal{F}, g(w) \vDash \psi$ and by induction hypothesis we get $(\mathcal{M}, w) \vDash \psi$. Hence by definition of g' above we have $p_\psi \in g'(w)$ and therefore $g'(w) = g(w) \cup \{p_\psi\} = s$. Second, suppose $p_\psi \notin s$. Then by $s \vDash (p_\psi \leftrightarrow \|\psi\|_\mathcal{F})$ we have $s \nvDash \|\psi\|_\mathcal{F}$. Note that p_ψ does not occur in $\|\psi\|_\mathcal{F}$, hence $t \nvDash \|\psi\|_\mathcal{F}$. By Theorem 1 we have $\mathcal{F}, g(w) \nvDash \psi$ and by induction hypothesis we get $(\mathcal{M}, w) \nvDash \psi$. Hence by definition of g' above we have $p_\psi \notin g'(w)$ and therefore $g'(w) = g(w) = s$. In both cases we have $g'(w) = s$.
For right to left, suppose $g'(w) = s$ for some w. By C3 for g we have that $t := g(w)$ is a state of $\mathcal{F}$, i.e. $t \vDash \theta$. Again we consider two cases. First, suppose $p_\psi \in s$. Then by definition of g' we have $M, w \vDash \psi$. Now by induction hypothesis we also have $(\mathcal{F}, g(w)) \vDash \psi$. By Theorem 1 we get $g(w) \vDash \|\psi\|_\mathcal{F}$. Note that p_ψ does not occur in $\|\psi\|_\mathcal{F}$. Hence we also have $s \vDash \|\psi\|_\mathcal{F}$. Second, suppose $p_\psi \notin s$. Then by definition of g' we have $M, w \nvDash \psi$. Now by induction hypothesis we also have $(\mathcal{F}, g(w)) \nvDash \psi$. By Theorem 1 we get $g(w) \nvDash \|\psi\|_\mathcal{F}$. Note that p_ψ does not occur in $\|\psi\|_\mathcal{F}$. Hence we also have $s \nvDash \|\psi\|_\mathcal{F}$. In both cases we have $s \vDash \theta \wedge (p_\psi \leftrightarrow \|\psi\|_\mathcal{F})$, i.e. s is a state of $\mathcal{F}_\psi^\Delta$.
Finally, by the induction hypothesis we have $(\mathcal{M}_\psi^\Delta, w) \vDash \xi$ iff $(\mathcal{F}_\psi^\Delta, g'(w)) \vDash \xi$ iff $(\mathcal{F}_\psi^\Delta, \{p_\psi\} \cup g(w)) \vDash \xi$. □

The following definition and theorem show that for every knowledge structure there is an equivalent Kripke model.

Definition 8. *For any $\mathcal{F} = (V, \theta, O_1, \ldots, O_n)$, we define the Kripke model $\mathcal{M}(\mathcal{F}) := (W, \pi, \mathcal{K}_1, \ldots, \mathcal{K}_n)$ as follows*

1. *W is the set of all states of $\mathcal{F}$,*
2. *for each $w \in W$, let the assignment $\pi(w)$ be w itself and*
3. *for each agent i and all $v, w \in W$, let $v \mathcal{K}_i w$ iff $v \cap O_i = w \cap O_i$.*

Theorem 2. *For any knowledge structure $\mathcal{F}$, any state s of $\mathcal{F}$, and any φ we have $(\mathcal{F}, s) \vDash \varphi$ iff $(M(\mathcal{F}), s) \vDash \varphi$.*

Proof. By Lemma 1 using the identity function as g. □

Example 6. We can apply Definition 8 to $\mathcal{F} = (\{p, q\}, p \to q, \{p\}, \{q\})$ from Example 2. The result is an equivalent Kripke model shown in Figure 7.

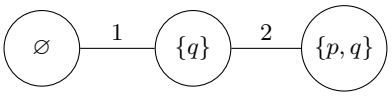


Fig. 7. Kripke model $\mathcal{M}(\mathcal{F})$ equivalent to $\mathcal{F}$ from Example 2.

Vice versa, for any S5 Kripke model we can find an equivalent knowledge structure. The essential idea is to add propositions as observational variables to encode the relations of each agent. To obtain a simple knowledge structure we should add as few propositions as possible. The method below adds $\sum_{i \in I} \mathsf{ceiling}(\log_2 k_i)$ propositions where k_i is the number of $\mathcal{K}_i$-equivalence classes. This could be further improved if one were to find a general way of using the propositions already present in the Kripke model as observational variables directly.

Definition 9. *For any S5 model $\mathcal{M} = (W, \pi, \mathcal{K}_1, \ldots, \mathcal{K}_n)$ with some set of primitive propositions U we define a knowledge structure $\mathcal{F}(\mathcal{M})$ as follows. For each agent i, write $\gamma_{i,1}, \ldots, \gamma_{i,k_i}$ for the equivalence classes given by $\mathcal{K}_i$ and let $l_i := \mathsf{ceiling}(\log_2 k_i)$. Let O_i be a set of l_i many fresh propositions. This yields the sets of observational variables $O_1, \ldots, O_n$, all disjoint to each other. If agent i has a total relation, i.e. only one equivalence class, then we have $O_i = \varnothing$. Enumerate k_i many subsets of O_i as $O_{\gamma_{i,1}}, \ldots, O_{\gamma_{i,k_i}}$ and define $g_i : W \to \mathcal{P}(O_i)$ by $g_i(w) := O_{\gamma_i(w)}$ where $\gamma_i(w)$ is the $\mathcal{K}_i$-equivalence class of w. Let $V := U \cup \bigcup_{0<i\leq n} O_i$ and define $g : W \to \mathcal{P}(V)$ by*

$$g(w) := \{v \in U \mid \pi(w)(v) = \top\} \cup \bigcup_{0<i\leq n} g_i(w)$$

Finally, let $\mathcal{F}(\mathcal{M}) := (V, \theta_M, O_1, \ldots, O_n)$ using

$$\theta_M := \bigvee \{g(w) \sqsubseteq V \mid w \in W\}$$

where $\sqsubseteq$ abbreviates a formula saying that out of the propositions in the second set exactly those in the first are true: $A \sqsubseteq B := \bigwedge A \wedge \bigwedge\{\neg p \mid p \in B \setminus A\}$.

Note that the idea here is to represent the state law θ_M as a BDD and not as a complex formula. Thereby we obtain a compact representation for many Kripke models, especially situations like the muddy children or blissful ignorance. However, in the worst case a BDD can have exponential size in the number of variables [7]. Hence $\mathsf{Bdd}(\theta_M)$ might be of size exponential in $|V|$.

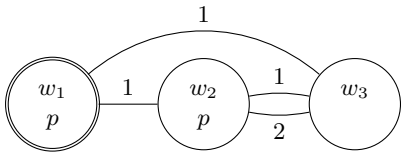
Theorem 3. *For any finite pointed S5 Kripke model $(\mathcal{M}, w)$ and every formula φ, we have that $(\mathcal{M}, w) \vDash \varphi$ iff $(\mathcal{F}(\mathcal{M}), g(w)) \vDash \varphi$.*

Proof. We have to check that Lemma 1 applies to Definition 9. To show C1, take any $w_1, w_2 \in W$ and $i \in \{1, \ldots, n\}$. Note that $g(w_1) \cap O_i = g(w_1) \cap O_i$ and $g(w_2) \cap O_i = g_i(w_2) \cap O_i$ because the observational variables introduced in Definition 9 are disjoint sets of fresh propositions. By definition of g_i we have that $g_i(w_1)$ and $g_i(w_2)$ are the same subset of O_i iff w_1 and w_2 are in the same $\mathcal{K}_i$-equivalence class. This shows that $g(w_1) \cap O_i = g(w_2) \cap O_i$ iff $w_1 \mathcal{K}_i w_2$.

For C2, take any $w \in W$ and any $v \in U$. Note that U is the original set of atomic propositions and therefore does not contain observational variables. Hence by definition of g we have $v \in g(w)$ iff $\pi(w)(v) = \top$.

For the "if" part of C3: If $s = g(w)$ for some $w \in W$, then by the definition of θ_M, we have that $g(w) \vDash \theta_M$ and hence $g(w)$ is a state of $\mathcal{F}(\mathcal{M})$. For the "only if" part, suppose s is a state of $\mathcal{F}(\mathcal{M})$. Then $s \vDash \theta_M$, hence it must satisfy one of the disjuncts and there must be a $w \in W$ such that $s \vDash g(w) \sqsubseteq V$. Now by definition of $\sqsubseteq$ we have $s = g(w)$. Now the theorem follows from Lemma 1. □

Example 7. Consider the pointed Kripke model $(\mathcal{M}, w_1)$ in Figure 8. Agent 2 knows that p, agent 1 does not know that p. Moreover, agent 1 does not even know whether agent 2 knows whether p.



$\mathcal{M}, w_1 \vDash K_2 p \wedge \neg K_1 p$
$\mathcal{M}, w_1 \vDash \neg K_1 (K_2 p \vee K_2 \neg p)$
$\mathcal{M}, w_1 \vDash \neg K_1 \neg (K_2 p \vee K_2 \neg p)$

Fig. 8. Pointed Kripke model $(\mathcal{M}, w_1)$ and some facts about it.

Now let us see how this knowledge and meta-knowledge get encoded symbolically. This direction is more difficult than going from a knowledge structure to a Kripke model as in Example 6, because here the worlds are not uniquely identified by valuations: $\pi(w_1) = \pi(w_2)$. Applying Definition 9 therefore means that we add one fresh proposition $O_2 := \{q\}$ to distinguish the two epistemic equivalence classes $\{w_1\}$ and $\{w_2, w_3\}$ of agent 2. For example, let $g_2(w_1) := \{q\}$ and $g_2(w_2) = g_2(w_3) := \varnothing$. Then we have $g(w_1) = \{p, q\}$, $g(w_2) = \{p\}$ and $g(w_1) = \varnothing$. Now we can compute the state law, a boolean formula over the vocabulary $V = \{p, q\}$, as follows:

$$\begin{aligned} \theta_M &= (g(w_1) \sqsubseteq V) \vee (g(w_2) \sqsubseteq V) \vee (g(w_3) \sqsubseteq V) \\ &= (\{p, q\} \sqsubseteq \{p, q\}) \vee (\{p\} \sqsubseteq \{p, q\}) \vee (\varnothing \sqsubseteq \{p, q\}) \\ &= (p \wedge q) \vee (p \wedge \neg q) \vee (\neg p \wedge \neg q) \\ &= q \to p \end{aligned}$$

The equivalent knowledge structure is thus

$$\mathcal{F}(\mathcal{M}) = (V' = \{p, q\}, \theta_M = q \to p, O_1 = \varnothing, O_2 = \{q\})$$

and the scene $(\mathcal{F}(\mathcal{M}), \{p, q\})$ is equivalent to $(\mathcal{M}, w_1)$.

7 Generalization to Action Models

What we have seen is how the two ways of modeling in this paper, though computationally different, are semantically equivalent. This leads us to consider how their interplay will work in more complex settings. The obvious direction to probe this is the area where DEL unleashes its full power: We now show how knowledge structures can be generalized to action models. Action models were first described in [2] and we do not repeat the basic definitions here but refer to [14] for a textbook treatment.

Definition 10. *An* action model *for a given vocabulary V and set of agents $I = \{1, \ldots, n\}$ is a tuple $\mathcal{A} = (A, \mathsf{pre}, R_1, \ldots, R_n)$ where A is a set of so-called* action points*, $\mathsf{pre} : A \to \mathcal{L}(V)$ assigns to each action point a formula called its* precondition *and $R_1, \ldots, R_n$ are binary relations on A. If all the relations are equivalence relations we call $\mathcal{A}$ an* S5 *action model.*

Given a Kripke model and an action model we define their product update *as $\mathcal{M} \times \mathcal{A} := (W', \pi', \mathcal{K}_1, \ldots, \mathcal{K}_n)$ where $W' := \{(w, \alpha) \in W \times A \mid \mathcal{M}, w \vDash \mathsf{pre}(\alpha)\}$, $\pi'((w, \alpha)) := \pi(w)$ and $(v, \alpha)\mathcal{K}'_i(w, \beta)$ iff $v\mathcal{K}_i w$ and $\alpha R_i \beta$.*

For any $\alpha \in A$ we call $(\mathcal{A}, \alpha)$ a pointed *(S5) action model.*

Example 8. Figure 9 shows a product update describing Example 1. The pointed S5 action model with two events announces p to agent 2 but not to agent 1. Still, agent 1 learns that 2 might have learned that p.

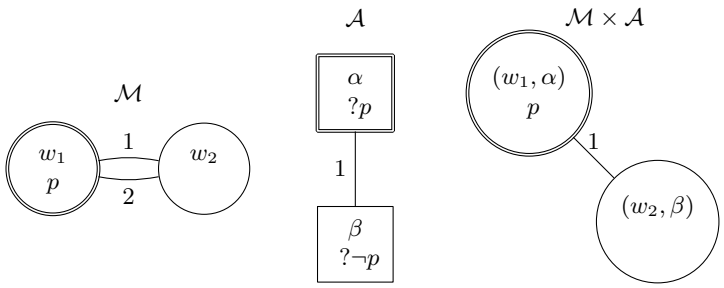


Fig. 9. Semi-Private announcement as a Product Update.

What action models are to Kripke models, the following knowledge transformers are to knowledge structures.

Definition 11. *A* knowledge transformer *for a given vocabulary V and set of agents $I = \{1, \ldots, n\}$ is a tuple $\mathcal{X} = (V^+, \theta^+, O_1, \ldots, O_n)$ where V^+ is a set of atomic propositions such that $V \cap V^+ = \varnothing$, θ^+ is a possibly epistemic formula from $\mathcal{L}(V \cup V^+)$ and $O_i \subseteq V^+$ for all agents i. An* event *is a knowledge transformer together with a subset $x \subseteq V^+$, written as $(\mathcal{X}, x)$.*

The knowledge transformation *of a knowledge structure* $\mathcal{F} = (V, \theta, O_1, \ldots, O_n)$ *with a knowledge transformer* $\mathcal{X} = (V^+, \theta^+, O_1^+, \ldots, O_n^+)$ *for* V *is defined by:*

$$\mathcal{F} \times \mathcal{X} := (V \cup V^+, \theta \wedge ||\theta^+||_{\mathcal{F}}, O_1 \cup O_1^+, \ldots, O_n \cup O_n^+)$$

Given a scene $(\mathcal{F}, s)$ *and an event* $(\mathcal{X}, x)$ *we define* $(\mathcal{F}, s) \times (\mathcal{X}, x) := (\mathcal{F} \times \mathcal{X}, s \cup x)$.

Here θ^+ is not restricted to be a boolean formula, just like preconditions of action models can be arbitrary formulas. Still, applying a knowledge transformer to a knowledge structure should again yield a knowledge structure with a boolean formula as the new state law. Hence, in Definition 11 we not just take the conjunction of θ and θ^+ but first localize θ^+ to $\|\theta^+\|_{\mathcal{F}}$. This formula will be equivalent on the previous, but not necessarily on the new structure.

For example, if the announced formula contains a K_i operator, then we rewrite it by quantifying over $V \setminus O_i$, not over $V \cup V^+ \setminus O_i \cup O_i^+$ as one might first think. The latter would yield boolean equivalents with respect to $\mathcal{F} \times \mathcal{X}$ whereas the former is with respect to $\mathcal{F}$. Compare this to the product update in Definition 10 where the preconditions are also evaluated on the model *before* the update.

The two kinds of announcements from above fit into the general framework of knowledge transformers as follows.

Example 9. The public announcement of φ is the event $((\varnothing, \varphi, \varnothing, \ldots, \varnothing), \varnothing)$. The semi-private announcement of φ to a group of agents Δ is given by $((\{p_\varphi\}, p_\varphi \leftrightarrow \varphi, O_1^+, \ldots, O_n^+), \{p_\varphi\})$ where $O_i^+ = \{p_\varphi\}$ if $i \in \Delta$ and $O_i^+ = \varnothing$ otherwise.

An obvious question about knowledge transformers is how they relate to action models, i.e. whether they describe the same or a different class of events. The answer is the same as for the relation between Kripke models and knowledge structures: For any S5 action model there is an equivalent transformer and vice versa. We can make this precise as follows, using very similar ideas as for Definitions 8 and 9 and then using Lemma 1.

Definition 12. *For any Knowledge Transformer* $\mathcal{X} = (V^+, \theta^+, O_1^+, \ldots, O_n^+)$ *we define an S5 action model* $\mathsf{Act}(\mathcal{X})$ *as follows. First, let the set of actions be* $A := \mathcal{P}(V^+)$*. Second, for any two actions* $\alpha, \beta \in A$*, let* $\alpha R_i \beta$ *iff* $\alpha \cap O_i^+ = \beta \cap O_i^+$*. Third, for any* α*, let* $\mathsf{pre}(\alpha) := \theta^+ \left(\frac{\alpha}{\top}\right) \left(\frac{V^+ \setminus \alpha}{\bot}\right)$*. Finally, let* $\mathsf{Act}(\mathcal{X}) := (A, (R_i)_{i \in I}, \mathsf{pre})$.

Definition 13. *The function* Trf *maps an S5 action model* $\mathcal{A} = (A, (R_i)_{i \in I}, \mathsf{pre})$ *to a transformer as follows. Let* P *be a finite set of fresh propositions such that there is an injective labeling function* $g : A \to \mathcal{P}(P)$ *and let*

$$\Phi := \bigwedge \{(g(a) \sqsubseteq P) \to \mathsf{pre}(a) \mid a \in A\}$$

where $\sqsubseteq$ *is the "out of" abbreviation from Definition 9. Now, for each* i*: Write* A/R_i *for the set of equivalence classes induced by* R_i*. Let* O_i^+ *be a finite set of fresh propositions such that there is an injective* $g_i : A/R_i \to \mathcal{P}(O_i^+)$ *and let*

$$\Phi_i := \bigwedge \left\{ (g_i(\alpha) \sqsubseteq O_i) \to \left(\bigvee_{a \in \alpha} (g(a) \sqsubseteq P) \right) \,\middle|\, \alpha \in A/R_i \right\}$$

Finally, define $\mathsf{Trf}(\mathcal{A}) := (V^+, \theta^+, O_1^+, \ldots, O_n^+)$ *where* $V^+ := P \cup \bigcup_{i \in I} P_i$ *and* $\theta^+ := \Phi \wedge \bigwedge_{i \in I} \Phi_i$.

In contrast to the translation in Definition 9 where θ_M could be represented as a BDD, here we can not do so with θ^+ as it might contain non-boolean operators in Φ. Still, before taking the last conjunction we can compute a smaller equivalent, of the purely boolean $\bigwedge_{i \in I} \Phi_i$.

Example 10. We can translate the product update from Example 8 to a knowledge transformation as follows. First note that in $\mathcal{M}$ both agents have a total relation, hence we do not have to add observational variables. The equivalent knowledge structure is just $\mathcal{F}(\mathcal{M}) = (\{p\}, \top, \varnothing, \varnothing)$. Now we use Definition 13 to obtain $\mathsf{Trf}(\mathcal{A})$. Choose the set $P = \{q\}$ where q is fresh and label the events of $\mathcal{A}$ by $g(\alpha) := \{q\}$ and $g(\beta) := \varnothing$. We then get $\Phi := (q \to p) \wedge (\neg q \to \neg p) = q \leftrightarrow p$. Agent 1 also has a total relation in $\mathcal{A}$, so we can choose $O_1^+ = \varnothing$ and $g_1(\alpha) := g_1(\beta) := \varnothing$. Note that $\varnothing \sqsubseteq \varnothing = \top$. Hence $\Phi_1 = (\top \to (q \vee \neg q)) = \top$. For agent 2 we need two labels, so let $O_2^+ := \{r\}$ where r is fresh, $g_2(\alpha) := \{r\}$ and $g_2(\beta) := \varnothing$. Then we get $\Phi_2 = (r \to q) \wedge (\neg r \to \neg q) = r \leftrightarrow q$. Putting it all together we get $\theta^+ = (q \leftrightarrow p) \wedge (r \leftrightarrow q)$ and thereby this transformer:

$$\mathsf{Trf}(\mathcal{A}) = (\{q, r\}, ((q \leftrightarrow p) \wedge (r \leftrightarrow q)), \varnothing, \{r\})$$

Finally, we can calculate the knowledge transformation $\mathcal{F}(\mathcal{M}) \times \mathsf{Trf}(\mathcal{A})$:

$$\begin{aligned} & (\{p\}, \top, \varnothing, \varnothing) \\ \times\ & (\{q, r\}, ((q \leftrightarrow p) \wedge (r \leftrightarrow q)), \varnothing, \{r\}) \\ =\ & (\{p, q, r\}, ((q \leftrightarrow p) \wedge (r \leftrightarrow q)), \varnothing, \{r\}) \end{aligned}$$

Observe that θ^+ makes q and p equivalent which makes this transformer somewhat redundant. As we mentioned already in Example 9, the semi-private announcement is given by a simpler transformer using only one fresh proposition, in this case $(\{q\}, (q \leftrightarrow p), \varnothing, \{q\})$. In general however, the distinction between those propositions linked to preconditions and those describing the observation is needed to translate more complex action models to knowledge transformers.

It remains to show that our definitions to go back and forth between action models and knowledge transformers are truthful in general.

Theorem 4. *For any scene* $(\mathcal{F}, s)$*, any event* $(\mathcal{X}, x)$ *and any formula* φ *over the vocabulary of* $\mathcal{F}$ *we have:*

$$(\mathcal{F}, s) \times (\mathcal{X}, x) \vDash \varphi \iff (\mathcal{M}(\mathcal{F}) \times \mathsf{Act}(\mathcal{X})), (s, x) \vDash \varphi$$

For any pointed S5 Kripke model $(\mathcal{M}, w)$*, any pointed S5 action model* $(\mathcal{A}, \alpha)$ *and any formula* φ *over the vocabulary of* $\mathcal{M}$ *we have:*

$$\mathcal{M} \times \mathcal{A}, (w, \alpha) \vDash \varphi \iff \mathcal{F}(\mathcal{M}) \times \mathsf{Trf}(\mathcal{A}), (g_{\mathcal{M}}(w) \cup g_{\mathcal{A}}(\alpha)) \vDash \varphi$$

where $g_{\mathcal{M}}$ *is from the construction of* $\mathcal{F}(\mathcal{M})$ *in Definition 8 and* $g_{\mathcal{A}}$ *is from the construction of* $\mathsf{Trf}(\mathcal{A})$ *in Definition 13.*

Proof. We use Lemma 1. For the first part, g needs to map worlds of $\mathcal{M}(\mathcal{F}) \times \mathsf{Act}(\mathcal{X})$ to states of $\mathcal{F} \times \mathcal{X}$. The former are pairs $(s, x) \in \mathcal{P}(V) \times \mathcal{P}(V^+)$, hence we define $g(s, x) := s \cup x$. For the second part, g should map worlds of $\mathcal{M} \times \mathcal{A}$ to states of $\mathcal{F}(\mathcal{M}) \times \mathsf{Trf}(\mathcal{A})$. Hence let $g(w, \alpha) := g_M(w) \cup g_{\mathcal{A}}(\alpha)$. It is straightforward to check C1 to C3 for both functions. □

To conclude this section, note that extensions of $\mathcal{L}(V)$ with dynamic operators for events still provide the advantage that all formulas have a local boolean equivalent. One only has to add a clause like this to Definition 6:

$$\|[\mathcal{X}, x]\varphi\|_{\mathcal{F}} := \|\theta^{+^*}\|_{\mathcal{F}} \to \|\varphi^*\|_{\mathcal{F} \times \mathcal{X}}$$

where $\theta^{+^*} := \theta^+ \left(\frac{x}{\top}\right) \left(\frac{V^+ \setminus x}{\bot}\right)$ and $\varphi^* := \varphi \left(\frac{x}{\top}\right) \left(\frac{V^+ \setminus x}{\bot}\right)$.

8 Generalization to non-S5 models

So far we only considered a "hard" notion of knowledge, characterized by the modal logic S5. Many other modalities and their dynamics can be formalized using DEL [14]. Belief for example is sometimes modeled using weaker modal logics with more general relational semantics using arbitrary relations. In this section we extend our symbolic framework to non-S5 models. To emphasize that the underlying relations do not have to be equivalence relations and we are no longer talking about knowledge, we now write $\mathcal{R}_i$ instead of $\mathcal{K}_i$ for the epistemic relations in a general Kripke model and $\Box_i$ instead of K_i for the modal operator in our language. Moreover, we change the semantics of group announcements $[\psi]_\Delta$ to be fully private, as the following definition and example show.

Definition 14. *Semantics for DEL on general Kripke models are defined inductively as in Definition 3 with the following changes.*

1. *For knowledge:*

$$(\mathcal{M}, s) \vDash \Box_i \varphi \quad \textit{iff} \quad \textit{for all } w' \in W : \textit{If } w \mathcal{R}_i w' \textit{ then } (\mathcal{M}, w') \vDash \varphi.$$

2. *For simplicity we do not interpret common knowledge on non-S5 models.*
3. *Public announcements are interpreted in the same way, i.e. restricting W.*
4. *Group announcements are now interpreted privately as follows: $(\mathcal{M}, w) \vDash [\psi]_\Delta \varphi$ iff $(\mathcal{M}, w) \vDash \psi$ implies that $(\mathcal{M}^\Delta_\psi, (w, 1)) \vDash \varphi$ where $\mathcal{M}^\Delta_\psi$ is a new Kripke model defined by*
   - $W^{\mathcal{M}^\Delta_\psi} := \{(w, 0) \mid w \in W\} \cup \{(w, 1) \mid w \in W \textit{ and } (\mathcal{M}, w) \vDash \psi\}$
   - $(w, b) \mathcal{R}_i^{\mathcal{M}^\Delta_\psi} (w', b')$ *iff* $w \mathcal{R}_i^M w'$ *and* $\begin{cases} b = b' \textit{ if } i \in \Delta \\ b' \neq 1 \textit{ otherwise} \end{cases}$
   - *and* $\pi^{\mathcal{M}^\Delta_\psi}(w, b) := \pi^M(w)$

Example 11. We modify Example 1: Suppose Alice reads the letter in private, such that Bob does not notice that she read it. This can be modeled as a private announcement of p to Alice as shown in Figure 10. Note that we now use directed

arrows for the epistemic accessibility relations as they now longer have to be equivalences.

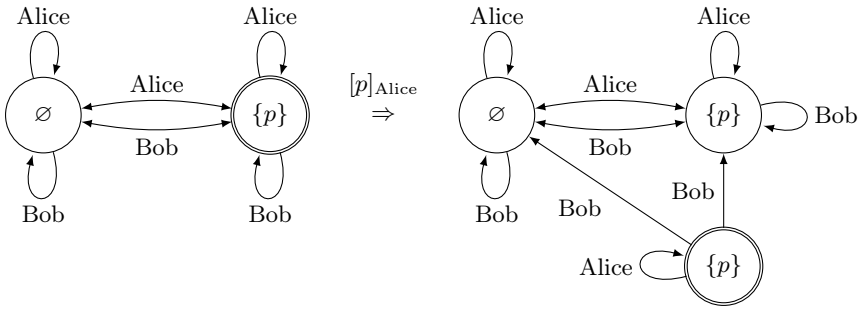


Fig. 10. Alice is secretly reading the letter.

To also enable symbolic model checking for these logics, we can use boolean encodings of relations which are widely used for model checking transition systems, starting with the seminal [11]. All states in our structures satisfy a unique set of propositions. Hence any relation over states is also a relation over sets of propositions. To encode these relations we use an idea from [28] where BDDs have also been used to model belief change: We replace the observational variables O_i with a BDD Ω_i for each agent. This BDD uses a double vocabulary: Suppose the knowledge structure is based on $V = \{p, q\}$, then each Ω_i encodes a boolean formula over $\{p, q, p', q'\}$. The formula encoded by Ω_i is true exactly for the *pairs of boolean assignments* that are connected by i's epistemic relation. For example if at state $\{p, q\}$ the agent deems the state $\{q\}$ possible, then $\{p, q, q'\}$ makes Ω_i true. Equivalently the opposite direction would be represented by $\{q, p', q'\}$ making Ω_i true. The following makes precise how we use this primed notation and the encoding of relations. More details and explanation can be found in [10, Section 5.2].

Definition 15. *If s is an assignment for V, then s' is the corresponding assignment for V'. For example, $\{p_1, p_3\}' = \{p_1', p_3'\}$. If φ is a boolean formula, $(\varphi)'$ is the result of priming all propositions in φ. For example, $(p_1 \to (p_3 \wedge \neg p_2))' = (p_1' \to (p_3' \wedge \neg p_2'))$. If s and t' are assignments for V and V' respectively such that $V \cap V' = \varnothing$ and φ is a formula over $V \cup V'$, we write $st' \vDash \varphi$ to say that $s \cup t'$ makes φ true. Given a relation R on $\mathcal{P}(V)$, we define the boolean formula*

$$\Phi(R) := \bigvee_{(s,t)\in R} ((s \sqsubseteq \mathsf{V}) \wedge (t \sqsubseteq \mathsf{V})')$$

where $s \sqsubseteq V$ is again the "out of" abbreviation from Definition 9. Note that this is a formula over $V \cup V'$ and we have that sRt iff $st' \vDash \Phi(R)$.

Example 12. Figures 11 to 13 show an example from [28, p. 136]. We start with a relation R over states with the vocabulary $V = \{p_1, p_2\}$. That is, $R \subseteq (\mathcal{P}(\{p_1, p_2\}))^2$. The formula $\Phi(R)$ shown in Figure 12 is a disjunction with one disjunct for each edge in the graph of R. We use V for the source and V' for the target. For example, the second disjunct $\neg p_1 \wedge \neg p_2 \wedge \neg p'_1 \wedge p'_2$ is for the edge from the top left state $\varnothing$ to the top right state $\{p_2\}$. In contrast, there is no R-edge from the top right to the bottom right state, hence the disjunct $\neg p_1 \wedge p_2 \wedge p'_1 \wedge p'_2$ is not in $\Phi(R)$. In our implementation the formula $\Phi(R)$ is never constructed explicitly. Instead we represent it using the BDD shown in Figure 13.

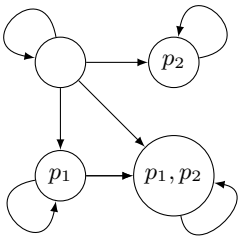


Fig. 11. Relation R.

$$
\begin{aligned}
&(\neg p_1 \wedge \neg p_2 \wedge \neg p'_1 \wedge \neg p'_2)\\
\vee\ &(\neg p_1 \wedge \neg p_2 \wedge \neg p'_1 \wedge p'_2)\\
\vee\ &(\neg p_1 \wedge \neg p_2 \wedge p'_1 \wedge \neg p'_2)\\
\vee\ &(\neg p_1 \wedge \neg p_2 \wedge p'_1 \wedge p'_2)\\
\vee\ &(\neg p_1 \wedge p_2 \wedge \neg p'_1 \wedge p'_2)\\
\vee\ &(p_1 \wedge \neg p_2 \wedge p'_1 \wedge p'_2)\\
\vee\ &(p_1 \wedge \neg p_2 \wedge p'_1 \wedge \neg p'_2)\\
\vee\ &(p_1 \wedge p_2 \wedge p'_1 \wedge p'_2)
\end{aligned}
$$

Fig. 12. Formula $\Phi(R)$.

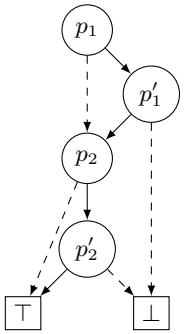


Fig. 13. BDD of $\Phi(R)$.

Implementing this encoding poses a few design choices. In particular we have to choose an initial variable ordering for the double vocabulary. For example we could stack all unprimed above the primed variables. But for our use cases the interleaving order $p_1, p'_1, p_2, p'_2, \ldots$ seems preferable because the epistemic relations of our agents are often at least partially decided by differences in a single variable. Hence each p should be close to its p' to keep the BDD small.

Definition 16. *Suppose we have n agents. A* belief structure *is a tuple $\mathcal{F} = (V, \theta, \Omega_1, \ldots, \Omega_n)$ where V is a finite set of propositional variables, θ is a boolean formula over V and for each agent i, Ω_i is a boolean formula over $V \cup V'$.*

Definition 17. *Semantics for DEL on scenes based on belief structures are defined inductively as in Definition 5 with the following changes.*

1. *For knowledge: $(\mathcal{F}, s) \vDash \Box_i \varphi$ iff for all $t \in \mathcal{F}$: If $st' \vDash \Omega_i$ then $(\mathcal{F}, t) \vDash \varphi$.*
2. *For simplicity we do not interpret common knowledge on belief structures.*
3. *Public announcements are still interpreted in the same way by restricting the state law of the belief structure.*
4. *However, we change the semantics of private announcements to make them fully private in the sense that all agents who are not in the group Δ will not notice anything: Let*

$$(\mathcal{F}, s) \vDash [\psi]_\Delta \varphi \text{ iff } (\mathcal{F}, s) \vDash \psi \Rightarrow (\mathcal{F}^\Delta_\psi, s \cup \{p_\psi\}) \vDash \varphi$$

where p_ψ is a new propositional variable, $\|\psi\|_{\mathcal{F}}$ is given by Definition 6 and

$$\mathcal{F}^{\Delta}_{\psi} := (V \cup \{p_\psi\}, \theta \wedge (p_\psi \to \|\psi\|_{\mathcal{F}}), \Omega^*_1, \ldots, \Omega^*_n)$$

where $\Omega^*_i := \Omega_i \wedge (p_\psi \leftrightarrow p'_\psi)$ if $i \in \Delta$ and $\Omega^*_i := \Omega_i \wedge \neg p'_\psi$ otherwise.

This already uses $\|\cdot\|_{\mathcal{F}}$ where $\mathcal{F}$ is now a belief structure. As on knowledge structures, all formulas have boolean equivalents with respect to a given belief structure. The translation in Definition 6 can be extended with

$$\|\Box_i \psi\|_{\mathcal{F}} := \forall V'(\theta' \to (\Omega_i \to (|\varphi|_{\mathcal{F}})'))$$

which due to the quantification over V' does not contain any primed propositions. To see that this translation preserves and reflects truth, add the following case to the proof of Theorem 1:

$$\begin{aligned}
\mathcal{F}, s \vDash \Box_i \varphi &\iff \text{For all } t \in \mathcal{F} : \text{If } st' \vDash \Omega_i \text{ then } (\mathcal{F}, t) \vDash \varphi \\
&\iff \text{For all } t \in \mathcal{F} : \text{If } st' \vDash \Omega_i \text{ then } t \vDash \|\varphi\|_{\mathcal{F}} \\
&\iff \text{For all } t : \text{If } t \in \mathcal{F} \text{ and } st' \vDash \Omega_i \text{ then } t \vDash \|\varphi\|_{\mathcal{F}} \\
&\iff \text{For all } t : \text{If } t \vDash \theta \text{ and } st' \vDash \Omega_i \text{ then } t \vDash \|\varphi\|_{\mathcal{F}} \\
&\iff \text{For all } t : \text{If } t' \vDash \theta' \text{ and } st' \vDash \Omega_i \text{ then } t' \vDash (\|\varphi\|_{\mathcal{F}})' \\
&\iff \text{For all } t : \text{If } st' \vDash \theta' \text{ and } st' \vDash \Omega_i \text{ then } st' \vDash (\|\varphi\|_{\mathcal{F}})' \\
&\iff \text{For all } t : st' \vDash \theta' \to (\Omega_i \to (\|\varphi\|_{\mathcal{F}})') \\
&\iff s \vDash \forall V' : (\theta' \to (\Omega_i \to (\|\varphi\|_{\mathcal{F}})'))
\end{aligned}$$

Belief structures are a generalization of knowledge structures: Any set of observational variables O can also be encoded using the BDD of the boolean formula $\Omega(O) := \bigwedge_{p \in O}(p \leftrightarrow p')$. This describes the same relation as O because for any two states s and t we have $st' \vDash \Omega(O)$ iff $s \cap O = t \cap O$.

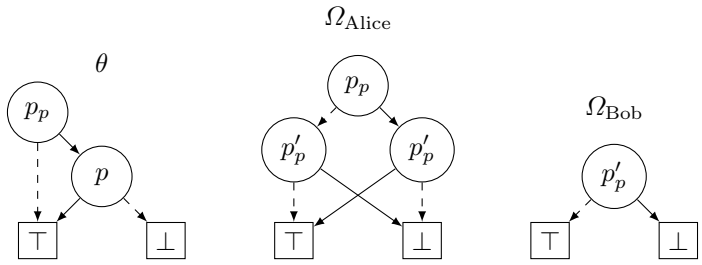
Example 13. We can also model Example 11 as a private announcement on belief structures. The initial structure is

$$\mathcal{F} = (V = \{p\}, \theta = \top, \Omega_{\text{Alice}} = \top, \Omega_{\text{Bob}} = \top)$$

and after the update we have

$$\mathcal{F}^{\text{Alice}}_{p} = (V = \{p, p_p\}, \theta = (p_p \to p), \Omega_{\text{Alice}} = (p_p \leftrightarrow p'_p), \Omega_{\text{Bob}} = \neg p'_p)$$

where the three boolean formulas can be represented using these BDDs:



We can see how this corresponds to the second Kripke model in Figure 10: First note that the state law θ is satisfied by the three states $\varnothing$, $\{p\}$ and $\{p, p_p\}$ which we can identify respectively with the worlds in the top left, top right and bottom. The observation law Ω_{Alice} then says that the upper and the lower part of the model are disconnected for Alice, whereas Bob almost has a total relation encoded in Ω_{Bob} up to the lower world being unreachable.

As the reader will already expect, such a correspondence between general Kripke models and belief structures can also be made precise. The following generalizes Lemma 1 from above. The only difference is in condition C1 dealing with the BDDs Ω_i now instead of variable sets O_i before.

Lemma 2. *Suppose for a set of agents $I = \{1, \dots, n\}$ we have a belief structure $\mathcal{F} = (V, \theta, \Omega_1, \dots, \Omega_n)$ and a finite Kripke model $M = (W, \pi, \mathcal{R}_1, \dots, \mathcal{R}_n)$ with a set of primitive propositions $U \subseteq V$. Furthermore, suppose we have a function $g : W \to \mathcal{P}(V)$ such that*

C1 For all $w_1, w_2 \in W$ and $i \in I$ we have that $g(w_1)(g(w_2)') \vDash \Omega_i$ iff $w_1 \mathcal{R}_i w_2$.
C2 For all $w \in W$ and $p \in U$, we have that $p \in g(w)$ iff $\pi(w)(p) = \top$.
C3 For every $s \subseteq V$, s is a state of $\mathcal{F}$ iff $s = g(w)$ for some $w \in W$.

Then, for every $\mathcal{L}(U)$-formula φ we have $(\mathcal{F}, g(w)) \vDash \varphi$ iff $(\mathcal{M}, w) \vDash \varphi$.

Proof. By induction on φ, the same as for Lemma 1 up to the following cases:

1. Knowledge: If φ is of the form $K_i\psi$, then by Definition 17, we have $(\mathcal{F}, g(w)) \vDash \Box_i\psi$ iff $(\mathcal{F}, s) \vDash \psi$ for all states s of $\mathcal{F}$ with $g(w)s' \vDash \Omega_i$. By C3 this is equivalent to having $(\mathcal{F}, g(w')) \vDash \psi$ for all $w' \in W$ with $g(w)g(w')' \vDash \Omega_i$, which by C1 is equivalent to $(\mathcal{F}, g(w')) \vDash \psi$ for all $w' \in W$ with $w\mathcal{R}_i w'$. Now by the induction hypothesis, this is equivalent to $(\mathcal{M}, w') \vDash \psi$ for all $w' \in W$ with $w\mathcal{R}_i w'$ which is exactly $(\mathcal{M}, w) \vDash \Box_i\psi$ by Definition 14.
2. Public announcements: Suppose φ is of the form $[\psi]_\Delta\xi$. As in the proof for Lemma 1 it suffices to show that $(\mathcal{M}^\psi, W) \vDash \xi$ iff $(\mathcal{F}^\psi, g(w)) \vDash \xi$. To do so, let g' be the restriction of g to $W^{\mathcal{M}^\psi} = \{w \in W \mid (\mathcal{M}, w) \vDash \psi\}$. It remains to show that g' fulfills C1 to C3. The new C1 is also a universal condition and holds for g on W^M, hence it must also hold for g' with respect to the restricted set $W^{\mathcal{M}^\psi} \subseteq W^M$. Conditions C2 and C3 are unchanged, hence the same proof applies. Together, g' fulfills all three conditions and by the induction hypothesis we get that $(\mathcal{M}^\psi, W) \vDash \xi$ iff $(\mathcal{F}^\psi, g(w)) \vDash \xi$.
3. Private announcements: Suppose φ is of the form $[\psi]_\Delta\xi$. By Definitions 14 and 17, we have that $(\mathcal{M}, w) \vDash [\psi]_\Delta\xi$ iff $(\mathcal{M}, w) \vDash \psi$ implies $(\mathcal{M}^\Delta_\psi, (w, 1)) \vDash \xi$, and $(\mathcal{F}, g(w)) \vDash [\psi]_\Delta\xi$ iff $(\mathcal{F}, g(w)) \vDash \psi$ implies $(\mathcal{F}^\Delta_\psi, \{p_\psi\} \cup g(w)) \vDash \xi$.
   As $(\mathcal{M}, w) \vDash \psi$ iff $(\mathcal{F}, g(w)) \vDash \psi$ by induction hypothesis, it suffices to prove that $(\mathcal{M}^\Delta_\psi, (w, 1)) \vDash \xi$ iff $(\mathcal{F}^\Delta_\psi, g(w) \cup \{p_\psi\}) \vDash \xi$.
   Let $g' : W^{\mathcal{M}^\Delta_\psi} \to \mathcal{P}(V \cup \{p_\psi\})$ be defined by $g'(w, b) := g(w) \cup \{p_\psi \mid b = 1\}$. By $g(w, 1) = g(w) \cup \{p_\psi\}$ and the induction hypothesis it remains to show that g' also fulfills C1 to C3.

For C1, take any $(w_1, b_1), (w_2, b_2) \in W^{\mathcal{M}^{\Delta}_{\psi}}$ and any i. As in Definitions 14 and 17 let $\mathcal{R}_i^{\mathcal{M}^{\Delta}_{\psi}}$ and Ω_i^* be the epistemic relations and the observation laws after the announcement. Consider two cases in which we have the following equivalences. The steps $\heartsuit$ follow from C1 with respect to g. First, if $i \in \Delta$:

$$\begin{aligned}
& (g'(w_1, b_1))(g'(w_2, b_2)') \vDash \Omega_i^* \\
\iff & (g(w_1) \cup \{p_\psi \mid b_1 = 1\})((g(w_2) \cup \{p_\psi \mid b_2 = 1\})') \vDash \Omega_i \wedge (p_\psi \leftrightarrow p'_\psi) \\
\iff & (g(w_1))(g(w_2)') \vDash \Omega_i \text{ and } b_1 = b_2 \\
\overset{\heartsuit}{\iff} & w_1 \mathcal{R}_i^M w_2 \text{ and } b_1 = b_2 \\
\iff & (w_1, b_1) \mathcal{R}_i^{\mathcal{M}^{\Delta}_{\psi}} (w_2, b_1)
\end{aligned}$$

Second, if $i \notin \Delta$:

$$\begin{aligned}
& (g'(w_1, b_1))(g'(w_2, b_2)') \vDash \Omega_i^* \\
\iff & (g(w_1) \cup \{p_\psi \mid b_1 = 1\})((g(w_2) \cup \{p_\psi \mid b_2 = 1\})') \vDash \Omega_i \wedge \neg p'_\psi \\
\iff & (g(w_1))(g(w_2)') \vDash \Omega_i \text{ and } b_2 \neq 1 \\
\overset{\heartsuit}{\iff} & w_1 \mathcal{R}_i^M w_2 \text{ and } b_2 \neq 1 \\
\iff & (w_1, b_1) \mathcal{R}_i^{\mathcal{M}^{\Delta}_{\psi}} (w_2, b_1)
\end{aligned}$$

To show C2, take any $(w, b) \in W^{\mathcal{M}^{\Delta}_{\psi}}$ and any $p \in U$. Note that $p_\psi \notin U$ and thus $p \neq p_\psi$. Hence we have $p \in g'(w, b)$ iff $p \in g(w)$. By C2 for g the latter is equivalent to $\pi^M(w)(p) = \top$ which by Definition 14 is equivalent to $\pi^{\mathcal{M}^{\Delta}_{\psi}}(w, b)(p) = \top$.

For C3, take any $s \subseteq V \cup \{p_\psi\}$. To show left to right, suppose s is a state of $\mathcal{F}^{\Delta}_{\psi}$. Then $s \vDash \theta \wedge (p_\psi \to \|\psi\|_{\mathcal{F}})$. In particular $s \vDash \theta$, so $t := s \cap V$ is a state of $\mathcal{F}$. Hence by C3 for g there is a $w \in W$ such that $g(w) = t$. We consider two cases. First, suppose $p_\psi \in s$. Then by $s \vDash (p_\psi \to \|\psi\|_{\mathcal{F}})$ we have $s \vDash \|\psi\|_{\mathcal{F}}$. Note that p_ψ does not occur in $\|\psi\|_{\mathcal{F}}$, hence $t \vDash \|\psi\|_{\mathcal{F}}$. By Theorem 1 we have $\mathcal{F}, g(w) \vDash \psi$ and by induction hypothesis we get $(\mathcal{M}, w) \vDash \psi$. Moreover, we have a world $(w, 1)$ in $\mathcal{M}^{\Delta}_{\psi}$ and by definition of g' above we have $p_\psi \in g'(w, 1)$. Therefore $g'(w, 1) = g(w) \cup \{p_\psi\} = s$. Second, suppose $p_\psi \notin s$. By Definition 14 we have a world $(w, 0)$ in $\mathcal{M}^{\Delta}_{\psi}$ and by definition of g' above we have $p_\psi \notin g'(w, 0)$. Therefore $g'(w, 0) = g(w) = s$. In both cases we found a world (w, b) in the updated model corresponding to s, i.e. such that $g'(w, b) = s$. For right to left, suppose $g'(w, b) = s$ for some $(w, b) \in W^{\mathcal{M}^{\Delta}_{\psi}}$. By C3 for g we have that $t := g(w)$ is a state of $\mathcal{F}$, i.e. $t \vDash \theta$. Again we consider two cases. First, suppose $p_\psi \in s$. Then by definition of g' we have $b = 1$. Hence by Definition 14 we must have $(\mathcal{M}, w) \vDash \psi$ and therefore by induction hypothesis also $(\mathcal{F}, g(w)) \vDash \psi$. By Theorem 1 with the addition on page 29 we get $g(w) \vDash \|\psi\|_{\mathcal{F}}$. Note that p_ψ does not occur in $\|\psi\|_{\mathcal{F}}$. Hence we also have $s \vDash \|\psi\|_{\mathcal{F}}$. Second, suppose $p_\psi \notin s$. Then we have $s \vDash (p_\psi \to \|\psi\|_{\mathcal{F}})$. In both cases we have $s \vDash \theta \wedge (p_\psi \to \|\psi\|_{\mathcal{F}})$, i.e. s is a state of $\mathcal{F}^{\Delta}_{\psi}$. Finally, by the induction hypothesis we have $(\mathcal{M}^{\Delta}_{\psi}, (w, 1)) \vDash \xi$ iff $(\mathcal{F}^{\Delta}_{\psi}, g'(w, 1)) \vDash \xi$ iff $(\mathcal{F}^{\Delta}_{\psi}, g(w) \cup \{p_\psi\}) \vDash \xi$.

□

We now also generalize the translation methods from Definitions 8 and 9. Lemma 2 then allows us to show their correctness and get generalized versions of Theorems 2 and 3: For every belief structure there is an equivalent Kripke model and vice versa.

Definition 18. *For any belief structure $\mathcal{F} = (V, \theta, \Omega_1, \ldots, \Omega_n)$, we define the Kripke model $\mathcal{M}(\mathcal{F}) := (W, \pi, \mathcal{R}_1, \ldots, \mathcal{R}_n)$ as follows*

1. *W is the set of all states of $\mathcal{F}$,*
2. *for each $w \in W$, let the assignment $\pi(w)$ be w itself and*
3. *for each agent i and all $w, w' \in W$, let $w\mathcal{R}_i w'$ iff $ww' \vDash \Omega_i$.*

Definition 19. *For any finite Kripke model $\mathcal{M} = (W, \pi, \mathcal{R}_1, \ldots, \mathcal{R}_n)$ we define a belief structure $\mathcal{F}(\mathcal{M})$ as follows. W.l.o.g. we assume unique valuations, i.e. that for all $w, w' \in W$ we have $\pi(w) \neq \pi(w')$. If this is not the case, we can add propositions to V and extend π in such a way that $\pi(w) \neq \pi(w')$. The maximum number of propositions we might have to add is $\mathsf{ceiling}(\log_2 |W|)$. Let $\mathcal{F}(\mathcal{M}) := (V, \theta_M, \Omega_1, \ldots, \Omega_n)$ where*

1. *V is the vocabulary on which $\mathcal{M}$ is interpreted with added propositions if necessary to make valuations unique,*
2. *$\theta_M := \bigvee\{s \sqsubseteq V \mid \exists w \in W : \pi(w) = s\}$ using $\sqsubseteq$ from Definition 9,*
3. *for each i the boolean formula $\Omega_i := \Phi(R_i)$ represents the relation R_i on $\mathcal{P}(V)$ given by $R_i st$ iff $\exists v, w \in W : \pi(v) = s \wedge \pi(w) = t \wedge \mathcal{R}_i vw$.*

Note that different from Definition 9 here we do not have to add propositions to distinguish all equivalence classes of all agents. This is because the Ω_is can carry more information than the simple sets of observed variables O_i.

Theorem 5. *For any belief structure $\mathcal{F}$, any state s of $\mathcal{F}$, and any φ we have $(\mathcal{F}, s) \vDash \varphi$ iff $(\mathcal{M}(\mathcal{F}), s) \vDash \varphi$.*

Proof. By Lemma 2 using the identity function as g. □

Theorem 6. *For any finite pointed Kripke model $(\mathcal{M}, w)$ and every formula φ, we have that $(\mathcal{M}, w) \vDash \varphi$ iff $(\mathcal{F}(\mathcal{M}), g(w)) \vDash \varphi$.*

Proof. We have to check that Lemma 2 applies to Definition 19. As we already assume unique valuations in $\mathcal{M}$, the appropriate injective function $g : W \to \mathcal{P}(V)$ is just defined by $g(w) := \{p \in V \mid \pi(w)(p) = \top\}$.

To show C1, take any $w_1, w_2 \in W$ and $i \in \{1, \ldots, n\}$. and note that $g(w_1)g(w_2)' \vDash \Omega_i$ iff $\pi(w_1)\pi(w_2)' \vDash \Phi(R_i)$ iff $w_1 \mathcal{R}_i w_2$.

For C2, take any $w \in W$ and any $v \in U$. By definition of g we have $v \in g(w)$ iff $\pi(w)(v) = \top$.

For the "if" part of C3: If $s = g(w)$ for some $w \in W$, then by the definition of θ_M, we have that $g(w) \vDash \theta_M$ and hence $g(w)$ is a state of $\mathcal{F}(\mathcal{M})$. For the "only if" part, suppose s is a state of $\mathcal{F}(\mathcal{M})$. Then $s \vDash \theta_M$, hence it must satisfy one of the disjuncts and there must be a $w \in W$ such that $s \vDash g(w) \sqsubseteq V$. Now by definition of $\sqsubseteq$ we have $s = g(w) = \pi(w)$.

Now the theorem follows from Lemma 2. □

Given this symbolic representation of Kripke models with arbitrary relations, one might wonder whether graph properties characterized by modal formulas also have corresponding BDD properties. The answer is positive and some examples are the following. The total relation is given by the constant $\top$ and the empty relation by $\bot$. To compute the inverse $\mathsf{Bdd}(R^{-1})$, simultaneously substitute primed for unprimed variables and vice versa in $\mathsf{Bdd}(R)$. The relation R is symmetric iff $\mathsf{Bdd}(R) = \mathsf{Bdd}(R^{-1})$. To get the symmetric closure, take $\mathsf{Bdd}(R) \vee \mathsf{Bdd}(R^{-1})$. Similarly, R is reflexive iff $\bigwedge_i (p_i \leftrightarrow p_i') \rightarrow \mathsf{Bdd}(R)$ is a tautology and the reflexive closure is given by $\mathsf{Bdd}(R) \vee \bigwedge_i (p_i \leftrightarrow p_i')$.

To check whether belief structures have similar computational advantages as knowledge structures we repeated the muddy children benchmark from Section 5 using the BDD encoding for relations instead of observational variables. As expected this worsens performance, but for larger cases of ten or more agents model checking on belief structures is still faster than DEMO-S5 — for example, it takes around 15 instead of 200 seconds to check the case of 12 agents. However, a better and more fair comparison would be with the original DEMO that can also handle non-S5 models and should be done with other scenarios than muddy children. We leave this as future work.

To conclude this section, note that this generalization is compatible with the one made in the previous section: One can define *belief transformers* in the same style as knowledge transformers in Definition 11, replacing the observed atomic propositions O_i^+ with BDDs Ω_i^+ encoding a relation on $\mathcal{P}(V^+)$. Thus we obtain a symbolic representation of events where observability need not be an equivalence relation, for example if someone is being deceived.

9 Conclusion and Future Work

We have achieved our goal of putting a new engine into DEL by a suitable semantic model transformation. This was shown to work well in various benchmarks, for example the Muddy Children and Russian cards. But there is obviously more to be explored now that we know this.

One line would be to use the same models with richer languages, and see whether the parallels that we found still persist. For example, factual change [4] should also be representable as knowledge transformers. They also motivate a new notion of action equivalence which might help to solve a problem with action models where bisimulation had to be replaced with the more complicated notion of action emulation [22].

Sections 7 and 8 showed that our framework can be generalized to cover many flavors of DEL, including action models and non-S5 notions like belief. Our benchmarks so far mainly concerned the S5 framework and we plan to explore the performance of the extensions in future work. As the Russian cards example (page 16) showed our model checker also performs well enough to deal with planning problems. While epistemic planning is often done using knowledge bases, our methods could bring logic back in the game.

As mentioned at the end of the last section we can combine knowledge transformers and the non-S5 encoding to obtain belief transformers. These will share both the features and the problems of non-S5 action models. As [20] says, "update of belief models with belief action models has a glitch": The result of updating a KD45 Kripke model with a KD45 action model does not have to be KD45. This is a good reason to study other models of belief which we have not discussed here, for example preference and plausibility orders. Finding a symbolic representation for these kinds of semantics was not in the scope of this paper but we hope to adapt our methods to them in the future.

On a technical side, we suspect that our program can still be further optimized to deal with much larger models. For example the usage of modern SAT solvers instead of BDDs could be interesting. Other abstraction ideas from the DEL literature could be implemented and their performance compared to our approach, for example the mental programs from [8].

But perhaps the deepest issue that we see emerging in our approach is this. While standard logical approaches to information flow assume a sharp distinction between syntax and semantic models, our BDD-oriented approach suggests the existence of a third intermediate level of representation combining features of both that may be the right level to be at, also from a cognitive viewpoint. We leave the exploration of the latter grander program to another occasion.

10 Acknowledgements

This work was partially supported by NSFC grant 61472369 and carried out within the Tsinghua-UvA Joint Research Center in Logic. We thank our anonymous referees for very useful comments and suggestions.

References

1. Aucher, G., Schwarzentruber, F.: On the complexity of dynamic epistemic logic. In: Proceedings of the 14th Conference on Theoretical Aspects of Rationality and Knowledge (TARK 2013), Chennai, India, January 7-9, 2013 (2013), http://www.tark.org/proceedings/tark_jan7_13/p19-aucher.pdf
2. Baltag, A., Moss, L.S., Solecki, S.: The logic of public announcements, common knowledge, and private suspicions. In: Bilboa, I. (ed.) TARK'98. pp. 43–56 (1998), http://dl.acm.org/citation.cfm?id=645876.671885
3. van Benthem, J., van Eijck, J., Gattinger, M., Su, K.: Symbolic model checking for dynamic epistemic logic. In: van der Hoek, W., Holliday, H.W., Wang, W.f. (eds.) Proceedings of The Fifth International Conference on Logic, Rationality and Interaction (LORI-V) in Taipei, Taiwan, October 28-30, 2015. pp. 366–378 (2015), http://dx.doi.org/10.1007/978-3-662-48561-3_30
4. van Benthem, J., van Eijck, J., Kooi, B.: Logics of communication and change. Information and computation 204(11), 1620–1662 (2006), http://dx.doi.org/10.1016/j.ic.2006.04.006
5. van Benthem, J., Gerbrandy, J., Hoshi, T., Pacuit, E.: Merging frameworks for interaction. Journal of Philosophical Logic 38(5), 491–526 (2009), http://dx.doi.org/10.1007/s10992-008-9099-x

6. Blackburn, P., de Rijke, M., Venema, Y.: Modal Logic. No. 53 in Cambridge Tracts in Theoretical Computer Science, CUP, Cambridge, UK (2001)
7. Bryant, R.E.: Graph-Based Algorithms for Boolean Function Manipulation. IEEE Transaction on Computers C-35(8), 677–691 (1986), http://dx.doi.org/10.1109/TC.1986.1676819
8. Charrier, T., Schwarzentruber, F.: Arbitrary public announcement logic with mental programs. In: Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems. pp. 1471–1479. IFAAMAS (2015), http://dl.acm.org/citation.cfm?id=2772879.2773340
9. Chaum, D.: The dining cryptographers problem: Unconditional sender and recipient untraceability. Journal of Cryptology 1(1), 65–75 (1988), http://dx.doi.org/10.1007/BF00206326
10. Clarke, E.M., Grumberg, O., Peled, D.A.: Model Checking. The MIT Press, Cambridge, Massachusetts, USA (1999)
11. Clarke, E.M., Grumberg, O., Long, D.E.: Model checking and abstraction. ACM transactions on Programming Languages and Systems 16(5), 1512–1542 (1994), http://dx.doi.org/10.1145/186025.186051
12. Cordón-Franco, A., van Ditmarsch, H., Fernández-Duque, D., Soler-Toscano, F.: A geometric protocol for cryptography with cards. Designs, Codes and Cryptography 74(1), 113–125 (2015), http://dx.doi.org/10.1007/s10623-013-9855-y
13. van Ditmarsch, H.: The russian cards problem. Studia Logica 75(1), 31–62 (2003), http://dx.doi.org/10.1023/A:1026168632319
14. van Ditmarsch, H., van der Hoek, W., Kooi, B.: Dynamic epistemic logic, vol. 1. Springer Heidelberg (2007), http://dx.doi.org/10.1007/978-1-4020-5839-4
15. van Ditmarsch, H., van der Hoek, W., van der Meyden, R., Ruan, J.: Model Checking Russian Cards. Electr. Notes Theor. Comput. Sci. 149(2), 105–123 (2006), http://dx.doi.org/10.1016/j.entcs.2005.07.029
16. van Ditmarsch, H., van der Hoek, W., Ruan, J.: Connecting dynamic epistemic and temporal epistemic logics. Logic Journal of IGPL 21(3), 380–403 (2013), http://dx.doi.org/10.1093/jigpal/jzr038
17. Duque, D.F., Goranko, V.: Secure aggregation of distributed information. CoRR abs/1407.7582 (2014), http://arxiv.org/abs/1407.7582
18. van Eijck, J.: DEMO—a demo of epistemic modelling. In: Interactive Logic. Selected Papers from the 7th Augustus de Morgan Workshop, London. vol. 1, pp. 303–362 (2007), http://homepages.cwi.nl/~jve/papers/07/pdfs/DEMO_IL.pdf
19. van Eijck, J.: DEMO-S5. Tech. rep., CWI (2014), http://homepages.cwi.nl/~jve/software/demo_s5
20. van Eijck, J.: Dynamic epistemic logics. In: Baltag, A., Smets, S. (eds.) Johan van Benthem on Logic and Information Dynamics, pp. 175–202. Springer (2014), http://dx.doi.org/10.1007/978-3-319-06025-5_7
21. van Eijck, J., Orzan, S.: Epistemic verification of anonymity. Electronic Notes in Theoretical Computer Science 168, 159–174 (2007), http://www.sciencedirect.com/science/article/pii/S1571066107000345, proceedings of the Second International Workshop on Views on Designing Complex Architectures (VODCA 2006)
22. van Eijck, J., Ruan, J., Sadzik, T.: Action emulation. Synthese 185(1), 131–151 (2012), http://dx.doi.org/10.1007/s11229-012-0083-1
23. Engesser, T., Bolander, T., Nebel, B.: Cooperative epistemic multi-agent planning with implicit coordination. Distributed and Multi-Agent Planning (DMAP-15) pp. 68–76 (2015), https://www.cs.bgu.ac.il/~icaps15/workshops/dmap2015-proceedings.pdf#page=72

24. Fagin, R., Halpern, J.Y., Moses, Y., Vardi, M.Y.: Reasoning about knowledge, vol. 4. MIT press Cambridge (1995)
25. Freudenthal, H.: Formulering van het 'som-en-product'-probleem. Nieuw Archief voor Wiskunde 17, 152 (1969)
26. Gattinger, M.: HasCacBDD version 0.1.0.0 (2016), https://github.com/m4lvin/HasCacBDD
27. Gierasimczuk, N., Szymanik, J.: A note on a generalization of the muddy children puzzle. In: Apt, K.R. (ed.) TARK'11. pp. 257–264. ACM (2011), http://dx.doi.org/10.1145/2000378.2000409
28. Gorogiannis, N., Ryan, M.D.: Implementation of Belief Change Operators Using BDDs. Studia Logica 70(1), 131–156 (2002), http://dx.doi.org/10.1023/A:1014610426691
29. Knuth, D.E.: The Art of Computer Programming. Combinatorial Algorithms, Part 1, vol. 4A. Addison-Wesley Professional (2011), https://cs.stanford.edu/~uno/taocp.html
30. Littlewood, J.: A Mathematician's Miscellany. Methuen, London (1953)
31. Lomuscio, A., Qu, H., Raimondi, F.: MCMAS: an open-source model checker for the verification of multi-agent systems. International Journal on Software Tools for Technology Transfer pp. 1–22 (2015), http://dx.doi.org/10.1007/s10009-015-0378-x
32. Lomuscio, A.R., van der Meyden, R., Ryan, M.: Knowledge in Multiagent Systems: Initial Configurations and Broadcast. ACM Trans. Comp. L. 1(2), 247–284 (2000), http://dx.doi.org/10.1145/359496.359527
33. Luo, X., Su, K., Sattar, A., Chen, Y.: Solving Sum and Product Riddle via BDD-Based Model Checking. In: Web Intel./IAT Workshops. pp. 630–633. IEEE (2008), http://dx.doi.org/10.1109/WIIAT.2008.277
34. Lv, G., Su, K., Xu, Y.: CacBDD: A BDD Package with Dynamic Cache Management. In: Proceedings of the 25th International Conference on Computer Aided Verification. pp. 229–234. CAV'13, Springer-Verlag, Berlin, Heidelberg (2013), http://dx.doi.org/10.1007/978-3-642-39799-8_15
35. van der Meyden, R., Su, K.: Symbolic Model Checking the Knowledge of the Dining Cryptographers. In: CSFW. pp. 280–. IEEE Computer Society (2004), http://dx.doi.org/10.1109/CSFW.2004.1310747
36. O'Sullivan, B.: Criterion (2016), http://www.serpentine.com/criterion
37. Somenzi, F.: CUDD: CU Decision Diagram Package Release 2.5.0 (2012), http://vlsi.colorado.edu/~fabio/CUDD/
38. Su, K., Sattar, A., Luo, X.: Model Checking Temporal Logics of Knowledge Via OBDDs. The Computer Journal 50(4), 403–420 (2007), http://dx.doi.org/10.1093/comjnl/bxm009

Appendix: Input and Output Examples

Based on the methods presented in this paper we implemented a model checker for DEL which can be used in two ways. First, similar to DEMO-S5 one can load SMCDEL as a Haskell module. This allows us to employ other Haskell functions and libraries and is especially useful to generate larger models and formulas automatically. All benchmarks in Section 5 were done this way.

Figure 14 shows how Muddy Children is encoded as a Kripke model for DEMO-S5. The function takes parameters n and m and returns the initial situation of n children out of which m are muddy. It makes heavy use of `bTables` which generates all possible boolean assignments for a set of propositions. Instead of using a valuation function the states itself are lists of boolean values that indicate which agents are muddy. Equivalence relations for each agent are then defined as partitions. We also list the output for the case of $n = m = 3$ and include a graph of the model in Figure 15.

```
mudDemoKrpInit :: Int -> Int -> DEMO_S5.EpistM [Bool]
mudDemoKrpInit n m = (DEMO_S5.Mo states agents [] rels points) where
  states = DEMO_S5.bTables n
  agents = map DEMO_S5.Ag [1..n]
  rels   = [(DEMO_S5.Ag i, [[tab1++[True]++tab2,tab1++[False]++tab2] |
                   tab1 <- DEMO_S5.bTables (i-1),
                   tab2 <- DEMO_S5.bTables (n-i) ]) | i <- [1..n] ]
  points = [replicate (n-m) False ++ replicate m True]
```

```
*Main> mudDemoKrpInit 3 3
Mo  [ [True ,True ,True ],[True ,True ,False],[True ,False,True ],
      [True ,False,False],[False,True ,True ],[False,True ,False],
      [False,False,True ],[False,False,False]]
    [Ag 1,Ag 2,Ag 3]
    []
    [(Ag 1,[[[True ,True ,True ],[False,True ,True ]],
            [[True ,True ,False],[False,True ,False]],
            [[True ,False,True ],[False,False,True ]],
            [[True ,False,False],[False,False,False]]])
    ,(Ag 2,[[[True ,True ,True ],[True ,False,True ]],
            [[True ,True ,False],[True ,False,False]],
            [[False,True ,True ],[False,False,True ]],
            [[False,True ,False],[False,False,False]]])
    ,(Ag 3,[[[True ,True ,True ],[True ,True ,False]],
            [[True ,False,True ],[True ,False,False]],
            [[False,True ,True ],[False,True ,False]],
            [[False,False,True ],[False,False,False]]])]
    [[True,True,True]]
```

Fig. 14. Muddy Children input and output for DEMO-S5.

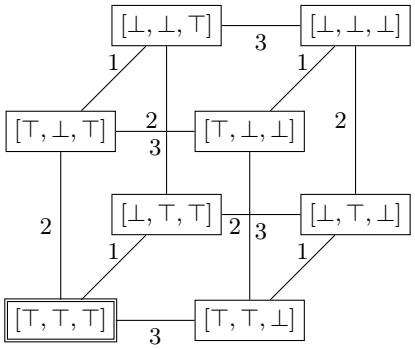


Fig. 15. Kripke model for Muddy Children.

In Figure 16 we list the function `mudScnInit` which takes the same parameters n and m but generates a knowledge structure for SMCDEL. The state law is simply $\top$ and each agent observes all but one proposition. Below the function we again list the example output for $n = m = 3$ and include a mathematical description of the structure. We can see that both the specification and the output are much shorter than their Kripke equivalents.

```
mudScnInit :: Int -> Int -> Scenario
mudScnInit n m = (KnS vocab law obs, actual) where
  vocab  = [P 1 .. P n]
  law    = boolBddOf Top
  obs    = [ (show i, delete (P i) vocab) | i <- [1..n] ]
  actual = [P 1 .. P m]
```

```
*Main> mudScnInit 3 3
(KnS [P 1,P 2,P 3] Top [("1",[P 2,P 3]),("2",[P 1,P 3]),("3",[P 1,P 2])]
    ,[P 1,P 2,P 3])
```

$$\left(V = \{p_1, p_2, p_3\}, \theta_0 = \top, \begin{array}{r} O_1 = \{p_2, p_3\} \\ O_2 = \{p_1, p_3\} \\ O_3 = \{p_1, p_2\} \end{array}\right), \{p_1, p_2, p_3\}$$

Fig. 16. Muddy Children input and output for SMCDEL.

To further simplify the usage of our model checker, we also provide an interface in which knowledge structures can be specified using a simple text format. In particular no knowledge of Haskell is needed here. An example input file for the Dining Cryptographers scenario with three agents is shown in Figure 17. We first describe the vocabulary in the `VARS` section. Then `LAW` contains a boolean formula, the state law. Under `OBS` we list which agent can observe what. After this we use `VALID?` and `WHERE?` followed by formulas. The former checks for validity

while the latter returns a list of states where the argument is true. Note that the indentation is just for readability. Whitespace and Haskell style comments are ignored by the program. The output can be printed to the command line as text (Figure 18) or as ready to use LaTeX code (Figure 19).

```
VARS
  0,     -- the NSA paid
  1,2,3, -- cryptographer i paid
  4,5,6  -- shared bits/coins

LAW -- exactly one cryptographer or the NSA paid
  AND ( OR (0,1,2,3), ~(0&1), ~(0&2), ~(0&3), ~(1&2), ~(1&3), ~(2&3) )

OBS
  alice: 1, 4,5
  bob  : 2, 4,  6
  carol: 3,   5,6

VALID?
  (alice,bob,carol) comknow that (OR (0,1,2,3))

WHERE?
  alice knows whether 0

VALID?
  [?! XOR (1, 4, 5)] -- After everyone announces the
  [?! XOR (2, 4, 6)] -- XOR of whether they paid and
  [?! XOR (3, 5, 6)] -- the coins they see ...
  AND (
    -- if the NSA paid this is common knowledge:
    0 -> (alice,bob,carol) comknow that 0,
    -- if one of the agents paid, the others don't know that:
    1 -> AND (~ bob knows that 1, ~ carol knows that 1),
    2 -> AND (~ bob knows that 1, ~ carol knows that 3),
    3 -> AND (~ bob knows that 1, ~ carol knows that 2)
  )
```

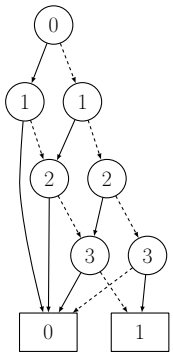
Fig. 17. Three Dining Cryptographers in SMCDEL.

```
Is Ck ["alice",...] (Disj [PrpF (P 0),...]) valid on the given structure?
True

Is Ck ["alice","bob","carol"] (Disj [...]) valid on the given structure?
True

At which states is Kw "alice" (PrpF (P 0)) true?
[1],[1,6],[1,5],[1,5,6],[1,4],[1,4,6],[1,4,5],[1,4,5,6]

Is PubAnnounceW (...) ... (Conj [...]) valid on the given structure?
True
```

Fig. 18. Output of SMCDEL on the command line (shortened).

Given Knowledge Structure

$$\mathcal{F} = \left(\{p, p_1, p_2, p_3, p_4, p_5, p_6\}, \quad \begin{array}{c} \{p_1, p_4, p_5\} \\ , \{p_2, p_4, p_6\} \\ \{p_3, p_5, p_6\} \end{array} \right)$$



Results

- Is $Ck_{\{alice,bob,carol\}} \bigvee \{p, p_1, p_2, p_3\}$ valid on $\mathcal{F}$?
  True.
- At which states is $K^{?}_{\text{alice}} p$ true?
  $\{p_1\}, \{p_1, p_6\}, \{p_1, p_5\}, \{p_1, p_5, p_6\}, \{p_1, p_4\}, \{p_1, p_4, p_6\}, \{p_1, p_4, p_5\}, \{p_1, p_4, p_5, p_6\}$
- Is $[?! \bigoplus \{p_1, p_4, p_5\}][?! \bigoplus \{p_2, p_4, p_6\}][?! \bigoplus \{p_3, p_5, p_6\}] \bigwedge \{(p \to Ck_{\{\text{a,b,c}\}} p), (p_1 \to (\neg K_{\text{b}} p_1 \wedge \neg K_{\text{c}} p_1)), (p_2 \to (\neg K_{\text{b}} p_1 \wedge \neg K_{\text{c}} p_3)), (p_3 \to (\neg K_{\text{b}} p_1 \wedge \neg K_{\text{c}} p_2))\}$ valid on $\mathcal{F}$?
  True.

Fig. 19. Output of SMCDEL in LaTeX.

.