Symbolic Model Checking for Dynamic Epistemic Logic

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

[1] ILLC, University of Amsterdam
[2] Department of Philosophy, Stanford University
[3] Centrum Wiskunde & Informatica, Amsterdam
[4] Institute for Integrated and Intelligent Systems, Griffith University
[5] Department of Computer Science, Jinan University

malvin@w4eg.eu

LORI-V, 2015-10-28, Taipei

Outline

Dynamic Epistemic Logic

Basic Definitions

Language

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_i\varphi \mid [!\varphi]\varphi$$

Kripke Models

$$\mathcal{M} = (W, R_i, V)$$

W	set of worlds
$R_i \subseteq W \times W$	indistinguishability relation
$V : W \to \mathcal{P}(P)$	valuation function

Semantics

- $\mathcal{M}, w \models K_i\varphi$ iff wR_iv implies $\mathcal{M}, v \models \varphi$
- $\mathcal{M}, w \models [!\varphi]\psi$ iff $\mathcal{M}, w \models \varphi$ implies $\mathcal{M}^{!\varphi}, w \models \psi$

(Yes, this is only PAL. We generalize later.)

Limits of Explicit Model Checking

Model Checking – The Task

Given a model and a formula, does it hold in the model:

$$\mathcal{M}, w \models \varphi \quad \text{or} \quad \mathcal{M}, w \not\models \varphi$$

???

Explicit Model Checking – the direct attack

Translate the semantics to ~~your favorite programming language~~ Haskell. First define what formulas and models are:

```
data Form = Top | PrpF Prp | Neg Form | Conj [Form]
  | K Agent Form | PubAnnounce Form Form

type State        = Int
type Partition    = [[State]]
type Assignment   = [(Prp,Bool)]
data KripkeModel = KrM [State]
                       [(Agent,Partition)]
                       [(State,Assignment)]
type PointModel   = (KripkeModel,State)
```

Explicit Model Checking – the direct attack

Then define $\models$ as a function:

```
eval :: PointModel -> Form      -> Bool
eval _                 Top       = True
eval (KrM _ _ val, w) (PrpF p)   = (val ! w) ! p
eval pm                (Neg f)   = not $ eval pm f
eval pm                (Conj fs) = all (eval pm) fs
eval (m@(KrM _ rel _),w) (K ag f) =
  all (\w' -> eval (m,w') f) vs where
    vs = concat $ filter (elem w) (apply rel ag)
eval pm (PubAnnounce form1 form2) =
  eval pm form1 ==> eval (pubAnnounce pm form1) form2
```

See *DEMO-S5* (Eijck 2014) for more elegance and details.

The Limits of Explicit Model Checking

- The set of possible worlds is explicitly constructed.
- Epistemic (equivalence) relations are spelled out.

$\Rightarrow$ Everything has to fit in memory.
For large models (1000 worlds) it gets slow.
Runtime in seconds for n Muddy Children:

n	DEMO-S5
3	0.000
6	0.012
8	0.273
10	8.424
11	46.530
12	228.055
13	1215.474

Symbolic Model Checking

Symbolic Model Checking

1. Can we represent models in a more compact way?
2. . . . such that we can still interpret all formulas?

Symbolic Model Checking

1. Can we represent models in a more compact way?
2. . . . such that we can still interpret all formulas?

There exist efficient methods for many temporal-epistemic logics (Su, Sattar, and Luo 2007).
Our contribution: How to do it for DEL.

Symbolic Model Checking

1. Can we represent models in a more compact way?
2. . . . such that we can still interpret all formulas?

There exist efficient methods for many temporal-epistemic logics (Su, Sattar, and Luo 2007).
Our contribution: How to do it for DEL.

1. Represent $\mathcal{M} = (W, R_i, V)$ symbolically: $\mathcal{F} = (V, \theta, O_i)$.
2. Translate DEL to equivalent boolean formulas.
3. Use BDDs to speed up boolean operations.

1. Knowledge Structures

Knowledge Structures

$$\mathcal{F} = (V, \theta, O_1, \cdots, O_n)$$

V	*Vocabulary*	a set of propositional variables
θ	*State Law*	a boolean formula over V
$O_i \subseteq V$	*Observables*	propositions observable by i

The set of states is $\{s \subseteq V \mid s \vDash \theta\}$. Call $(\mathcal{F}, s)$ a scenario.

1. Knowledge Structures

Knowledge Structures

$$\mathcal{F} = (V, \theta, O_1, \cdots, O_n)$$

V	*Vocabulary*	a set of propositional variables
θ	*State Law*	a boolean formula over V
$O_i \subseteq V$	*Observables*	propositions observable by i

The set of states is $\{s \subseteq V \mid s \vDash \theta\}$. Call $(\mathcal{F}, s)$ a scenario.

New Semantics:

$\mathcal{F}, s \vDash K\varphi$ iff $s \cap O_i = s' \cap O_i$ implies $\mathcal{F}, s' \vDash \varphi$

$\mathcal{F}, s \vDash [!\varphi]\psi$ iff $\mathcal{F}, s \vDash \varphi$ implies $\mathcal{F}^{\varphi}, s \vDash \psi$

1. Knowledge Structures

Knowledge Structures

$$\mathcal{F} = (V, \theta, O_1, \cdots, O_n)$$

V	*Vocabulary*	a set of propositional variables
θ	*State Law*	a boolean formula over V
$O_i \subseteq V$	*Observables*	propositions observable by i

The set of states is $\{s \subseteq V \mid s \vDash \theta\}$. Call $(\mathcal{F}, s)$ a scenario.

New Semantics:

$\mathcal{F}, s \vDash K\varphi$ iff $s \cap O_i = s' \cap O_i$ implies $\mathcal{F}, s' \vDash \varphi$

$\mathcal{F}, s \vDash [!\varphi]\psi$ iff $\mathcal{F}, s \vDash \varphi$ implies $\mathcal{F}^\varphi, s \vDash \psi$

where $\mathcal{F}^\varphi$ has the new state law $> \theta \wedge \|\varphi\|_\mathcal{F}$.

1. Knowledge Structures

Example

$$\mathcal{F} = (V = \{p\}, \theta = \top, O_1 = \{p\}, O_2 = \varnothing)$$

States: $\varnothing$, $\{p\}$
Some facts:

- $\mathcal{F}, \varnothing \models \neg p \wedge K_1 \neg p \wedge \neg K_2 \neg p$
- $\mathcal{F}, \{p\} \models p \wedge K_1 p \wedge \neg K_2 p$
- $\mathcal{F}, \{p\} \models [!p] K_2 p$

because:

$$\mathcal{F}^p = (V = \{p\}, \theta = p, O_1 = \{p\}, O_2 = \varnothing)$$

From Knowledge Structures to Kripke Models

Theorem: For every knowledge structure $\mathcal{F}$ there is an equivalent S5 Kripke Model $\mathcal{M}$ such that $\mathcal{F}, s \vDash \varphi$ iff $\mathcal{M}, w_s \vDash \varphi$.

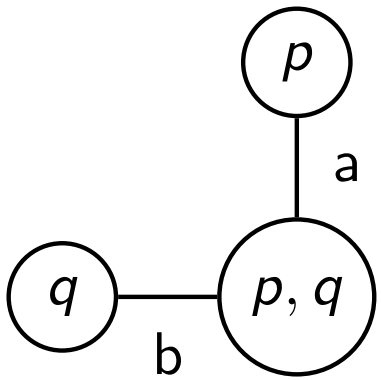
Proof.

Let $W := \{s \subseteq V \mid s \vDash \theta\}$, $V = \text{id}$ and $R_i st$ iff $s \cap O_i = t \cap O_i$.

Example: The knowledge structure

$$\mathcal{F} = (V = \{p, q\}, \theta = p \vee q, O_a = \{p\}, O_b = \{q\})$$

is equivalent to this Kripke model:



From Kripke Models to Knowledge Structures

This direction is non-trivial.

Theorem: For every S5 Kripke Model $\mathcal{M}$ there is an equivalent knowledge structure $\mathcal{F}$ such that $\mathcal{M}, w \models \varphi$ iff $\mathcal{F}, s_w \models \varphi$.

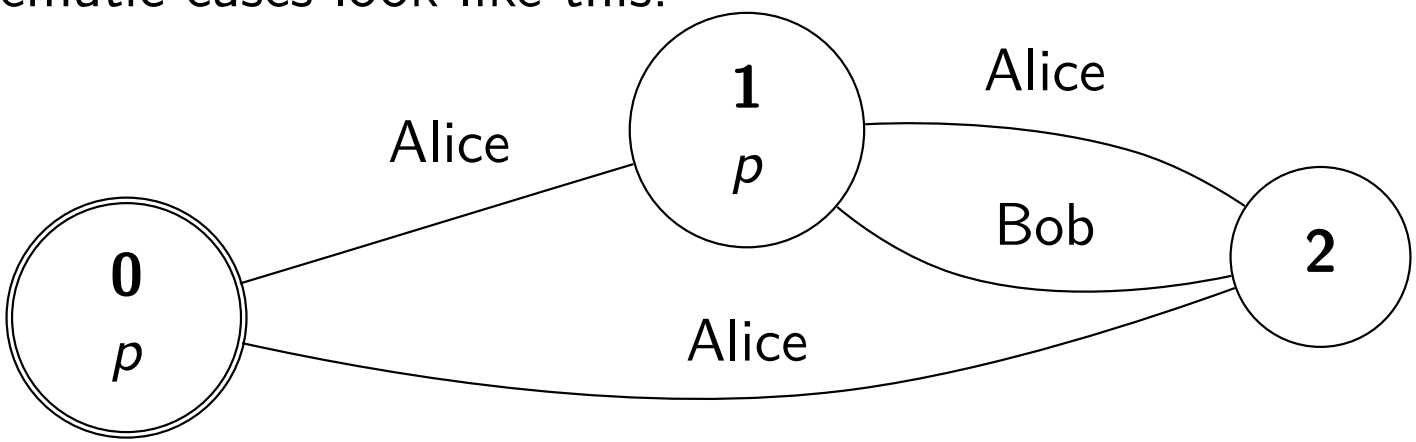
From Kripke Models to Knowledge Structures

This direction is non-trivial.

Theorem: For every S5 Kripke Model $\mathcal{M}$ there is an equivalent knowledge structure $\mathcal{F}$ such that $\mathcal{M}, w \vDash \varphi$ iff $\mathcal{F}, s_w \vDash \varphi$.

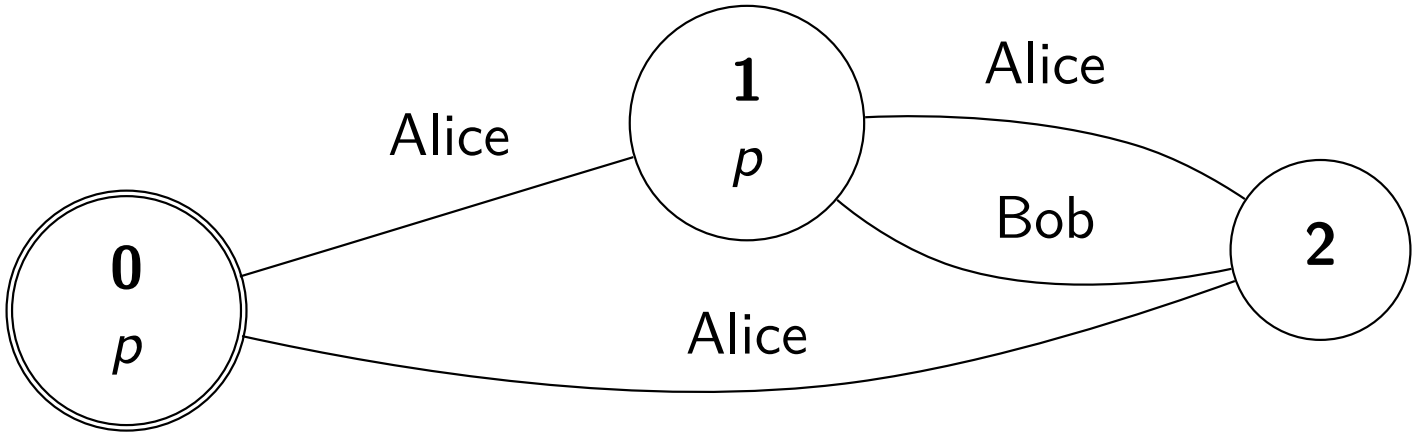
Proof.

Problematic cases look like this:



From Kripke Models to Knowledge Structures

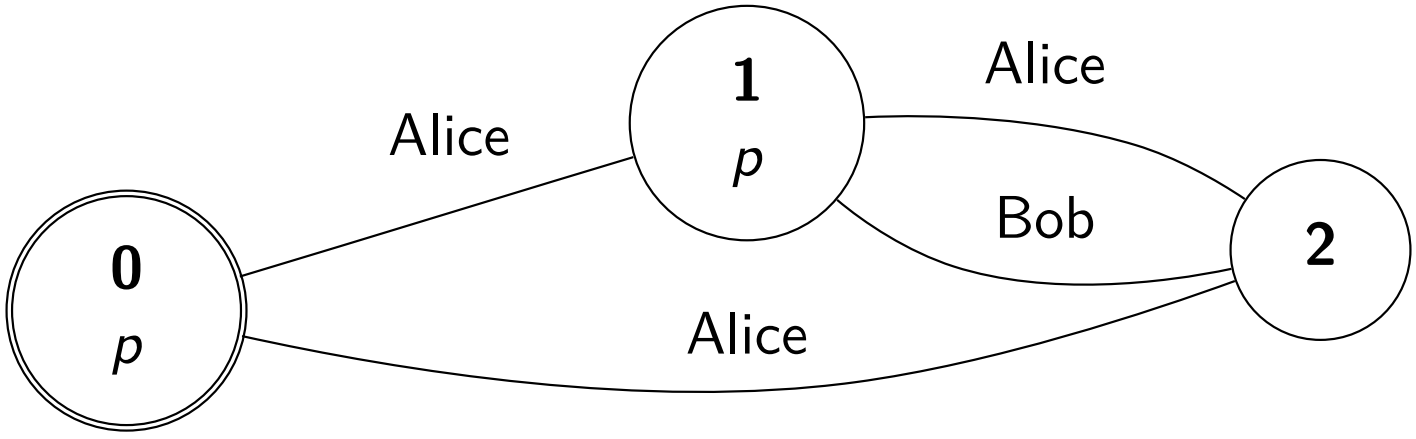
Proof. (continued)



Trick: Add propositions to distinguish all equivalence classes.

From Kripke Models to Knowledge Structures

Proof. (continued)

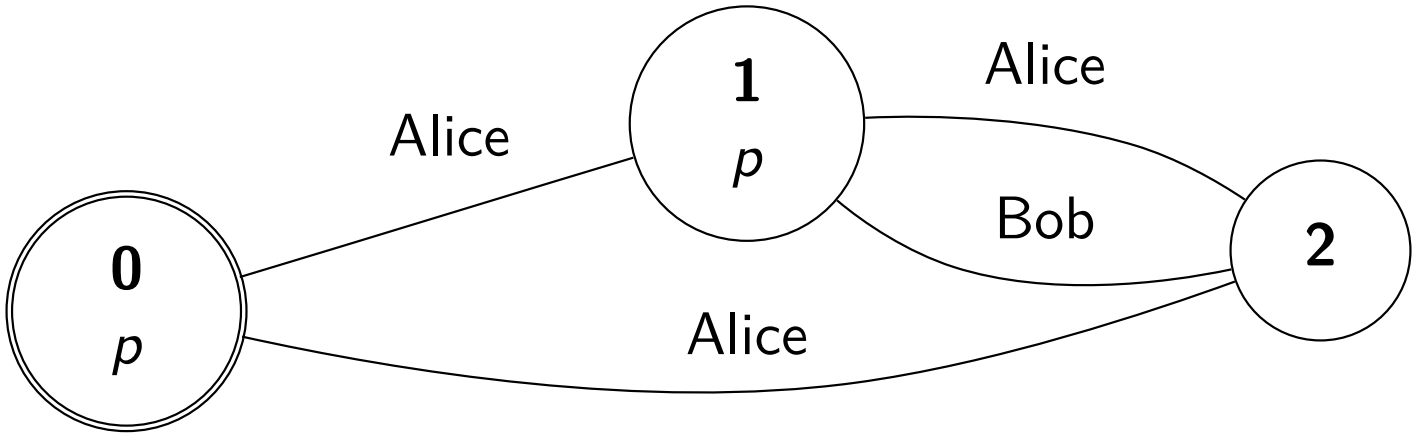


Trick: Add propositions to distinguish all equivalence classes.

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

From Kripke Models to Knowledge Structures

Proof. (continued)



is equivalent to

$$(V = \{p, p_2\},\ \theta = p_2 \to p,\ O_{\text{Alice}} = \varnothing,\ O_{\text{Bob}} = \{p_2\})$$

actual state: $\{p, p_2\}$

□

So what, Kripke Models and knowledge structures are the same?!

2. Everything is boolean!

Definition: Fix a knowledge structure $\mathcal{F} = (V, \theta, O_1, \cdots, O_n)$.
We translate everything to boolean formulas $\| \cdot \|_{\mathcal{F}}$:

p	p
$\neg\varphi$	$\neg\|\varphi\|_{\mathcal{F}}$
$\varphi_1 \wedge \varphi_2$	$\|\varphi_1\|_{\mathcal{F}} \wedge \|\varphi_2\|_{\mathcal{F}}$
$K_i\varphi$	$\forall(V \setminus O_i)(\theta \to \|\varphi\|_{\mathcal{F}})$
$[!\varphi]\psi$	$\|\varphi\|_{\mathcal{F}} \to \|\psi\|_{\mathcal{F}^\varphi}$

Theorem: For all scenarios $(\mathcal{F}, s)$ and all formulas φ:

$$\mathcal{F}, s \vDash \varphi \iff s \vDash \|\varphi\|_{\mathcal{F}}$$

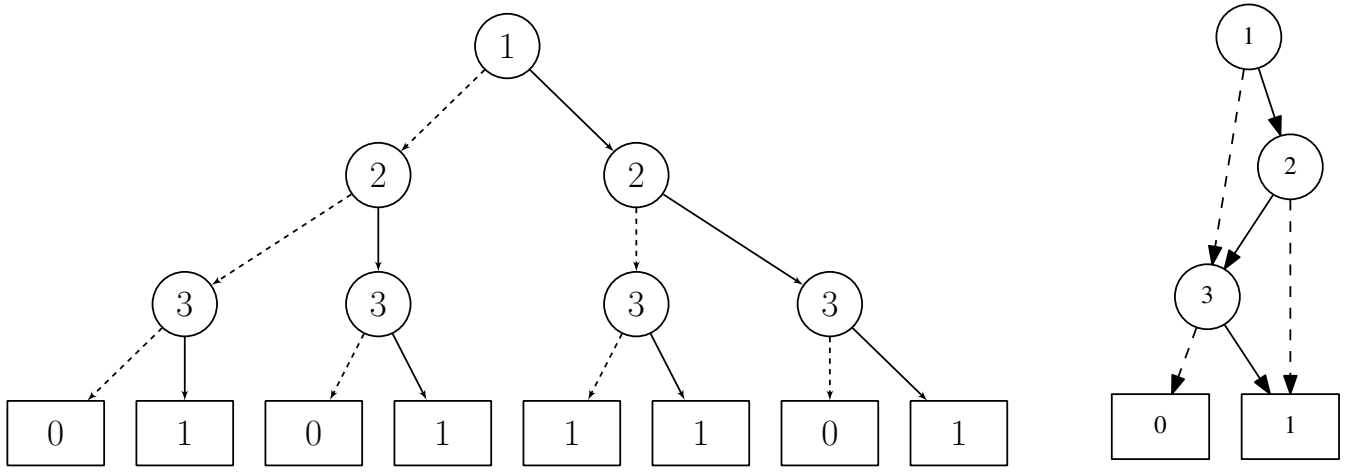
Why care about boolean formulas?

Binary Decision Diagrams

Truth Tables are dead, long live trees

Definition: A Binary Decision Diagram for the variables V is a directed acyclic graph where non-terminal nodes are from V with two outgoing edges and terminal nodes are $\top$ or $\bot$.

- All boolean functions can be represented like this.
- Ordered: Variables in a given order, maximally once.
- Reduced: No redundancy, identify isomorphic subgraphs.
- By "BDD" we always mean an ordered and reduced BDD.



(Read the classic Bryant 1986 for more details.)

BDD Magic

How long do you need to compare these two formulas?

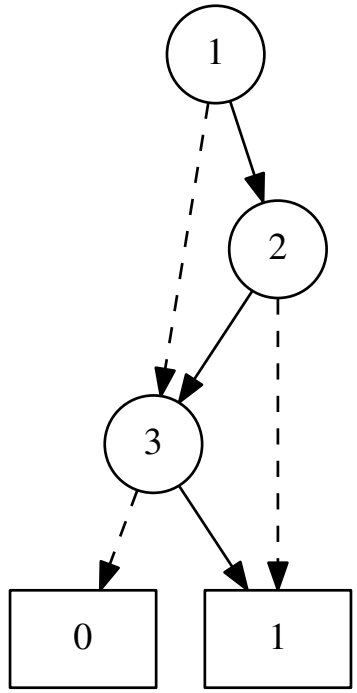
$$p_3 \vee \neg(p_1 \rightarrow p_2) \quad \equiv \quad \neg(p_1 \wedge \neg p_2) \rightarrow p_3$$

BDD Magic

How long do you need to compare these two formulas?

$$p_3 \vee \neg(p_1 \rightarrow p_2) \quad \equiv \quad \neg(p_1 \wedge \neg p_2) \rightarrow p_3$$

Here ~~are~~ is their BDD~~s~~:



BDD Magic

This was not an accident, BDDs are canonical.

Theorem:

$$\varphi \equiv \psi \quad \Rightarrow \quad \mathrm{BDD}(\varphi) = \mathrm{BDD}(\psi)$$

Equivalence checks are free and we have fast algorithms to compute $\mathrm{BDD}(\neg\varphi)$, $\mathrm{BDD}(\varphi \wedge \psi)$, $\mathrm{BDD}(\varphi \rightarrow \psi)$ etc.

The Plan

Given Kripke model $(\mathcal{M}, w)$ and DEL formula φ,

1. Represent $\mathcal{M}$ as a knowledge structure $\mathcal{F}$.
2. Compute the boolean equivalent $|\varphi|_{\mathcal{F}}$.
3. Ask a BDD package whether $w \vDash |\varphi|_{\mathcal{F}}$.

We use CacBDD (Lv, Su, and Xu 2013).

Examples and Results

Symbolic Muddy Children

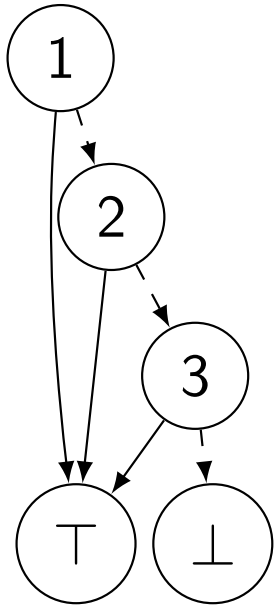
Initial knowledge structure:

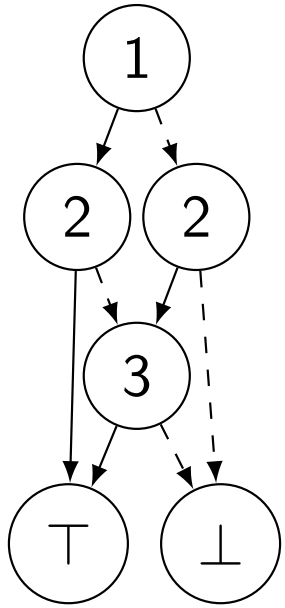
$$\mathcal{F} = (\{p_1, p_2, p_3\}, \top, O_1 = \{p_2, p_3\}, O_2 = \{p_1, p_3\}, O_3 = \{p_1, p_2\})$$

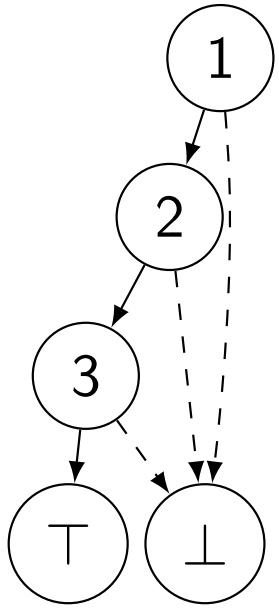
After the third announcement the children know their own state:

$$\varphi = [!(p_1 \vee p_2 \vee p_3)][! \bigwedge_i \neg(K_i p_i \vee K_i \neg p_i)][! \bigwedge_i \neg(K_i p_i \vee K_i \neg p_i)](\bigwedge_i (K_i p_i))$$

Intermediate BDDs for the state law:







Muddy Children

Runtime in seconds:

n	DEMO-S5	SMCDEL
3	0.000	0.000
6	0.012	0.002
8	0.273	0.004
10	8.424	0.008
11	46.530	0.011
12	228.055	0.015
13	1215.474	0.019
20		0.078
40		0.777
60		2.563
80		6.905

Further Results

- Sum and Product: 2 seconds to find the solution.
- Russian Cards: 4 seconds to find all 102 safe announcements.
- Dining Cryptographers: 10 seconds for the case of 160 agents.

Generalization to Action Models

Action Models

Action Model:

$$\mathcal{A} = (A, S_i, \text{pre})$$

A	set of actions
$S_i \subseteq A \times A$	indistinguishability relation
$\text{pre} : A \to \mathcal{L}$	preconditions

Product Update:

$\mathcal{M} \otimes \mathcal{A} := (W', R', V')$ where

- $W' = \{(w, a) \in W \times A \mid \mathcal{M}, w \vDash \text{pre}(a)\}$
- $R_i'(s, a)(t, b)$ iff $R_i st$ and $S_i ab$
- $V'(w, a) = V(w)$ no factual change

Semantics:

$\mathcal{M}, w \vDash [\mathcal{A}, a]\varphi$ iff $\mathcal{M}, w \vDash \text{pre}(a)$ implies $\mathcal{M} \otimes \mathcal{A}, (w, a) \vDash \varphi$

Knowledge Transformers

Knowledge Transformer:

$$\mathcal{X} = (V^+, \mu, O_1^+, \ldots, O_n^+)$$

V^+	*New Vocabulary*	new propositional variables
μ	*Event Law*	a formula over $V \cup V^+$
$O_i^+ \subseteq V^+$	*Observables*	what can i observe?

Transformation: Given $\mathcal{F} = (V, \theta, O_1, \ldots, O_n)$ and $\mathcal{X} = (V^+, \mu, O_1^+, \ldots, O_n^+)$, define

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

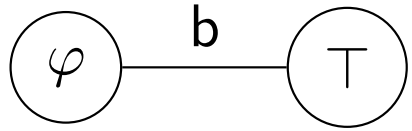
Event: $(\mathcal{X}, x)$ where $x \subseteq V^+$

Knowledge Transformers

Examples:

- public announcement: $\mathcal{X} = (\varnothing, \varphi, \varnothing, \varnothing)$
- (almost) private announcement of φ to a:

$$\mathcal{X} = (\{p\}, p \to \varphi, O_a = \{p\}, O_b = \varnothing)$$



Theorem: For every S5 action model $\mathcal{A}$ there is a transformer $\mathcal{X}$ (and vice versa) such that for every equivalent $\mathcal{M}$ and $\mathcal{F}$:

$$\mathcal{M} \otimes \mathcal{A}, (w, a) \vDash \varphi \iff \mathcal{F} \otimes \mathcal{X}, s \cup x \vDash \varphi$$

Conclusion

Summary

- Symbolic Model Checking DEL
  - represent Kripke Models with knowledge structures
  - translate formulas to boolean equivalents
  - use Binary Decision Diagrams for boolean reasoning
- The new software beats existing model checkers.

Future Work

Extensions of the framework:

- non-S5
- factual change

Optimization and alternatives:

- parallelization
- minimize structures
- use SAT instead of BDD

Theoretical questions:

- complexity of DEL fragments, puzzles, etc.
- BDDs as information states
- equivalence of actions / knowledge transformers
- symbolic model checking for other modal logics

Thank you for your attention!

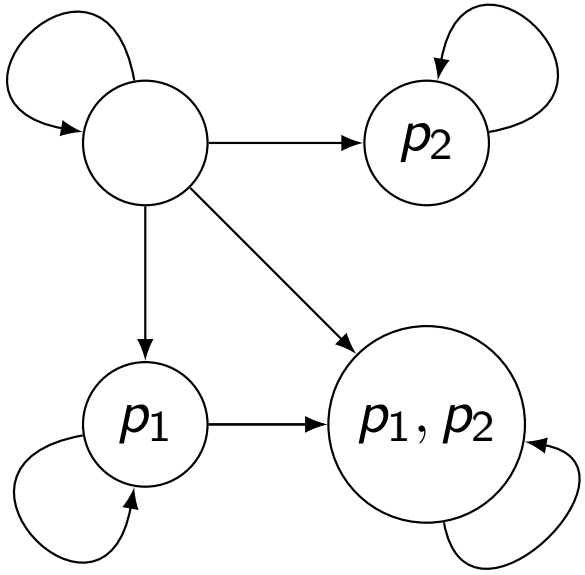
Contact: malvin@w4eg.eu

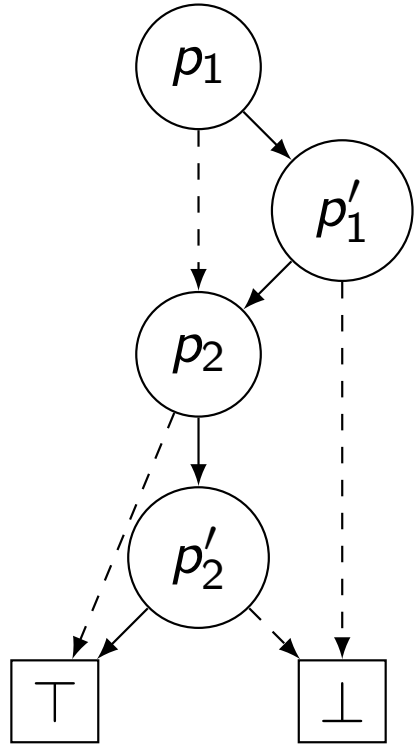
Try our software: github.com/jrclogic/SMCDEL

```
-- Three Muddy Children
VARS     1,2,3
LAW      Top
OBS      alice: 2,3
         bob:   1,3
         carol: 1,2
VALID?   ( not (alice knows whether 1)
         & not (bob   knows whether 2)
         & not (carol knows whether 3) )
WHERE?   [ ! (1|2|3) ]
         ( (alice knows whether 1)
         | (bob   knows whether 2)
         | (carol knows whether 3) )
```

Bonus slide 1: non-S5

We can replace O_i with a BDD Ω_i to describe any relation.
Trick: Use copy-propositions to describe reachable worlds.
(Gorogiannis and Ryan 2002)





Now translate $K_i\varphi$ to $\forall\vec{p'}(\theta' \rightarrow (\Omega_i(\vec{p}, \vec{p'}) \rightarrow (\|\varphi\|_{\mathcal{F}})'))$

Bonus slide 2: Comparing DEL and ETL

Many scenarios can be formalized in temporal logics and in DEL, e.g. the Dining Dryptographers by Chaum.
SMCDEL can check this quickly, but there are many questions:

- When are they equivalent? (Benthem et al. 2009, Ditmarsch, Hoek, and Ruan (2013))
- Which formalizations are more intuitive?
- What is faster
  - for your computer to model check?
  - for you to write down formulas?

References

Benthem, Johan van, Jelle Gerbrandy, Tomohiro Hoshi, and Eric Pacuit. 2009. "Merging Frameworks for Interaction." *Journal of Philosophical Logic* 38 (5). Springer: 491–526. doi:http://doi.org/10.1007/s10992-008-9099-x.

Bryant, Randal E. 1986. "Graph-Based Algorithms for Boolean Function Manipulation." *IEEE Transaction on Computers* C-35 (8): 677–91. doi:http://doi.org/10.1109/TC.1986.1676819.

Ditmarsch, Hans van, Wiebe van der Hoek, and Ji Ruan. 2013. "Connecting Dynamic Epistemic and Temporal Epistemic Logics." *Logic Journal of IGPL* 21 (3). Oxford Univ Press: 380–403. doi:http://doi.org/10.1093/jigpal/jzr038.

Eijck, Jan van. 2014. "DEMO-S5." CWI. http://www.cwi.nl/~jve/software/demo_s5.

Gorogiannis, Nikos, and Mark D. Ryan. 2002. "Implementation of Belief Change Operators Using BDDs." *Studia Logica* 70 (1). Kluwer Academic Publishers: 131–56. doi:http://doi.org/10.1023/A:1014610426691.

Lv, Guanfeng, Kaile Su, and Yanyan Xu. 2013. "CacBDD: A BDD Package with Dynamic Cache Management." In *Proceedings of the 25th International Conference on Computer Aided Verification*, 229–34. CAV'13. Berlin, Heidelberg: Springer-Verlag. doi:http://doi.org/10.1007/978-3-642-39799-8_15.

Su, Kaile, Abdul Sattar, and Xiangyu Luo. 2007. "Model Checking Temporal Logics of Knowledge Via OBDDs." *The Computer Journal* 50 (4): 403–20. doi:http://doi.org/10.1093/comjnl/bxm009.