Epistemic Modelling and Protocol Dynamics

Yanjing Wang

Department of Philosophy, Peking University

Oct. 26, 2010

A classic example

Muddy Children - the setting

- Out of n children, $k \geq 1$ got mud on their foreheads while playing.

A classic example

Muddy Children - the setting

- Out of n children, $k \geq 1$ got mud on their foreheads while playing.
- They can see whether other kids are dirty, but there is no mirror for them to discover whether they are dirty themselves.

A classic example

Muddy Children - the setting

- Out of *n* children, $k \geq 1$ got mud on their foreheads while playing.
- They can see whether other kids are dirty, but there is no mirror for them to discover whether they are dirty themselves.
- Then father walks in and states: “At least one of you is dirty!” Then he requests “If you know you are dirty, step forward now.”

A classic example

Muddy Children - the setting

- Out of n children, $k \geq 1$ got mud on their foreheads while playing.
- They can see whether other kids are dirty, but there is no mirror for them to discover whether they are dirty themselves.
- Then father walks in and states: “At least one of you is dirty!” Then he requests “If you know you are dirty, step forward now.”
- If nobody steps forward, he repeats his request: “If you now know you are dirty, step forward now.”

A classic example

Muddy Children - the setting

- Out of n children, $k \geq 1$ got mud on their foreheads while playing.
- They can see whether other kids are dirty, but there is no mirror for them to discover whether they are dirty themselves.
- Then father walks in and states: "At least one of you is dirty!" Then he requests "If you know you are dirty, step forward now."
- If nobody steps forward, he repeats his request: "If you now know you are dirty, step forward now."
- After exactly k requests to step forward, the k dirty children suddenly do so (assuming they are honest and perfect reasoners).

A perfect logic for explaining the puzzle?

A perfect logic for explaining the puzzle?

The language of *Public Announcement Logic* (PAL [Pla89, GG97]) is defined as follows:

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

A perfect logic for explaining the puzzle?

The language of *Public Announcement Logic* (PAL [Pla89, GG97]) is defined as follows:

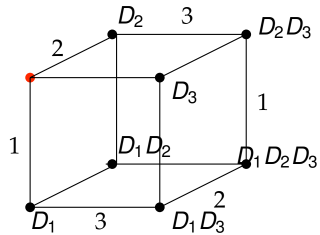
$$\phi ::= p \mid \phi \wedge \phi \mid \neg\phi \mid K_i\phi \mid [!\phi]\phi$$

It is interpreted on S5 models $\mathcal{M} = (S, \{\sim_i\}_{i\in\mathbf{I}}, V)$:

$$\begin{array}{|rcl|} \mathcal{M}, s \vDash K_i\phi & \Leftrightarrow & \text{for all } t \text{, if } s \sim_i t \text{ then } \mathcal{M}, t \vDash \phi \\ \mathcal{M}, s \vDash [!\psi]\phi & \Leftrightarrow & \text{if } \mathcal{M}, s \vDash \psi \text{ then } \mathcal{M}|_\psi, s \vDash \phi \end{array}$$

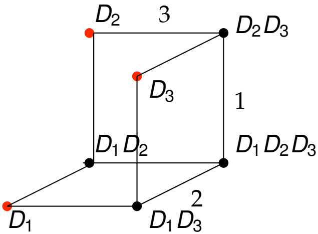
where $\mathcal{M}|_\psi = (S', \{\sim'_i\}_{i\in\mathbf{I}}, V')$ with $S' = \{s \in S \mid \mathcal{M}, s \vDash \psi\}$, $\sim'_i = \sim_i \cap (S' \times S')$, and $V' = V|_{S'}$

When there are 3 dirty children...



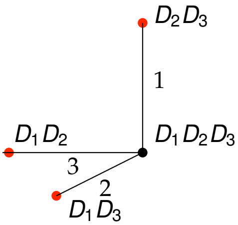
"At least one of you is dirty!"
Announcement: $\psi_0 = D_1 \vee D_2 \vee D_3$

When there are 3 dirty children...



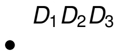
No one steps forward.
Announcement: $\psi_1 = \neg K_1 D_1 \wedge \neg K_2 D_2 \wedge \neg K_3 D_3$

When there are 3 dirty children...



No one steps forward.
Announcement: $\psi_1 = \neg K_1 D_1 \wedge \neg K_2 D_2 \wedge \neg K_3 D_3$

When there are 3 dirty children...



Now all the children know that they are dirty.

$\mathcal{M}, (D_1 D_2 D_3) \vDash [!\psi_0][\psi_1][\psi_1] K_1 D_1 \wedge K_2 D_2 \wedge K_3 D_3$

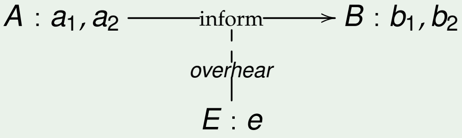
Is it really perfect?

Is it really perfect?

- Where was the father after the first announcement?

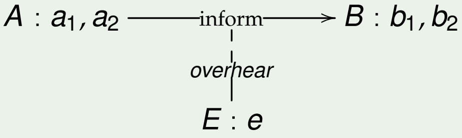
Is it really perfect?

Example (Procedural information is important)



Is it really perfect?

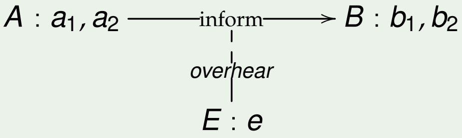
Example (Procedural information is important)



A “promising protocol” for this scenario is that A announces the disjunction of his actual hand (say 01) with all the different combinations of the remaining cards, so he would announce “I have 01 or 23 or 24 or 34.”

Is it really perfect?

Example (Procedural information is important)



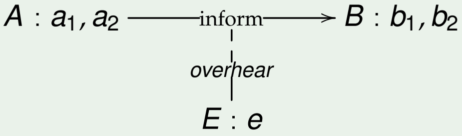
A "promising protocol" for this scenario is that A announces the disjunction of his actual hand (say 01) with all the different combinations of the remaining cards, so he would announce "I have 01 or 23 or 24 or 34."

It can go wrong if we assume...

- The goal of the protocol is commonly known.
- The procedure to generate the announcement is commonly known.

Is it really perfect?

Example (Procedural information is important)



A "promising protocol" for this scenario is that *A* announces the disjunction of his actual hand (say 01) with all the different combinations of the remaining cards, so he would announce "I have 01 or 23 or 24 or 34."

"If you know the protocol and it is assumed to be correct, then it may turn incorrect!"

Is it really perfect?

- Where was the father?

Is it really perfect?

- Where was the father?
- How to build a suitable model?

Is it really perfect?

- Where was the father?
- How to build a suitable model?
- What if there are a lot of children?

Is it really perfect?

- Dynamic (epistemic) logics with protocol announcements

Is it really perfect?

- Dynamic (epistemic) logics with protocol announcements
- Epistemic modelling

Is it really perfect?

- Dynamic (epistemic) logics with protocol announcements
- Epistemic modelling
- Epistemic abstraction

Is it really perfect?

- Dynamic (epistemic) logics with protocol announcements
- Epistemic modelling
- Epistemic abstraction

Protocols and their functions

Protocols and their functions

Protocols: in a very broad sense

Procedural rules that govern our everyday life

Protocols and their functions

Protocols: in a very broad sense

Procedural rules that govern our everyday life

Functions of protocols

- Let us know what to do (do *a* **then** do *b* or *c*).

Protocols and their functions

Protocols: in a very broad sense

Procedural rules that govern our everyday life

Functions of protocols

- Let us know what to do (do *a* **then** do *b* or *c*).
- Let us know the meaning of actions (**if** *p* **then** do a).

Protocols and their functions

Francois De La Rochefoucauld

True love is like ghosts, which everybody talks about and few have seen.

Protocols and their functions

Francois De La Rochefoucauld

True love is like ghosts, which everybody talks about and few have seen.

However...

You can actually show this ghost without seeing it or understanding what it is.

Protocol dynamics

This is how we did it in the past

Protocol dynamics

Until something evil came in...

Häagen-Dazs: “Love her, take her to Häagen-Dazs”.

Protocol dynamics

Until something evil came in...

Häagen-Dazs: “Love her, take her to Häagen-Dazs”.

Protocol dynamics

Until something evil came in...

Häagen-Dazs: “Love her, take her to Häagen-Dazs”.

The announcement of the slogan makes it commonly known that buying an icecream shows your love.

Protocol dynamics

Even “better”: a true “father” can change his mind.

Protocol dynamics

Current situation

Protocol announcement logic

Previous research

Protocols are not specified in the logical languages in the existing work of Dynamic Epistemic Logic and Epistemic Temporal Logic [HY09, vBGHP09, Hos09, HF89, PR03]

Protocol announcement logic

Previous research

Protocols are not specified in the logical languages in the existing work of Dynamic Epistemic Logic and Epistemic Temporal Logic [HY09, vBGHP09, Hos09, HF89, PR03]

To specify protocols and their dynamics explicitly!

Our logics are based on:

Protocol announcement logic

Previous research

Protocols are not specified in the logical languages in the existing work of Dynamic Epistemic Logic and Epistemic Temporal Logic [HY09, vBGHP09, Hos09, HF89, PR03]

To specify protocols and their dynamics explicitly!

Our logics are based on:

- Propositional Dynamic Logic (PDL [FL79])
- Public Announcement Logic (PAL [Pla89, GG97])

Protocol announcement logic

The first language $\mathtt{PDL}^!$

The formulas of $\mathtt{PDL}^!$ are built from a set of basic proposition letters **P** and a finite set of atomic action symbols **Σ** as follows:

$$\begin{array}{rcl} \phi & ::= & \top \mid p \mid \neg\phi \mid \phi \wedge \phi \mid [\pi]\phi \mid [!\pi]\phi \\ \pi & ::= & \mathbf{1} \mid \mathbf{0} \mid a \mid \pi \cdot \pi \mid \pi + \pi \mid \pi^* \end{array}$$

Protocol announcement logic

The first language $\texttt{PDL}^!$

The formulas of $\texttt{PDL}^!$ are built from a set of basic proposition letters **P** and a finite set of atomic action symbols $\mathbf{\Sigma}$ as follows:

$$\begin{array}{lll} \phi & ::= & \top \mid p \mid \neg\phi \mid \phi \wedge \phi \mid [\pi]\phi \mid [!\pi]\phi \\ \pi & ::= & \mathbf{1} \mid \mathbf{0} \mid a \mid \pi \cdot \pi \mid \pi + \pi \mid \pi^* \end{array}$$

The language of regular expressions

$$\mathcal{L}(\mathbf{0}) = \emptyset \qquad \mathcal{L}(\mathbf{1}) = \{\epsilon\} \qquad \mathcal{L}(a) = \{a\}$$

$$\mathcal{L}(\pi \cdot \pi') = \{wv \mid w \in \mathcal{L}(\pi), v \in \mathcal{L}(\pi')\}$$

$$\mathcal{L}(\pi + \pi') = \mathcal{L}(\pi) \cup \mathcal{L}(\pi')$$

$$\mathcal{L}(\pi^*) = \{\epsilon\} \cup \bigcup_{n>0}(\mathcal{L}(\underbrace{\pi \cdots \pi}_{n}))$$

Protocol announcement logic

The first language $\mathtt{PDL}^!$

The formulas of $\mathtt{PDL}^!$ are built from a set of basic proposition letters **P** and a finite set of atomic action symbols **Σ** as follows:

$$\begin{array}{lcl} \phi & ::= & \top \mid p \mid \neg\phi \mid \phi \wedge \phi \mid [\pi]\phi \mid [!\pi]\phi \\ \pi & ::= & \mathbf{1} \mid \mathbf{0} \mid a \mid \pi \cdot \pi \mid \pi + \pi \mid \pi^* \end{array}$$

Example (The language of regular expressions)

$(a + (b \cdot c))^* = \{\epsilon, a, bc, abc, aaa, bcbca \ldots\}$

The semantics

input derivative: $\backslash w$

The language of the $\pi\backslash w$ of a regular expression π is defined as $\mathcal{L}(\pi\backslash w) = \{v \mid wv \in \mathcal{L}(\pi)\}$.

The semantics

input derivative: $\backslash w$

The language of the $\pi\backslash w$ of a regular expression π is defined as $\mathcal{L}(\pi\backslash w) = \{v \mid wv \in \mathcal{L}(\pi)\}$. We can axiomatize the operation $\backslash w$ (cf. [Brz64, Con71]).

The semantics

input derivative: $\backslash w$

The language of the $\pi\backslash w$ of a regular expression π is defined as $\mathcal{L}(\pi\backslash w) = \{v \mid wv \in \mathcal{L}(\pi)\}$. We can axiomatize the operation $\backslash w$ (cf. [Brz64, Con71]).We say $w \in \mathbf{\Sigma}^*$ is *compliant with* π (notation: $w \propto \pi$) if $\pi\backslash w \neq \mathbf{0}$.

The semantics

input derivative: $\backslash w$

The language of the $\pi\backslash w$ of a regular expression π is defined as $\mathcal{L}(\pi\backslash w) = \{v \mid wv \in \mathcal{L}(\pi)\}$. We can axiomatize the operation $\backslash w$ (cf. [Brz64, Con71]).We say $w \in \mathbf{\Sigma}^*$ is *compliant with* π (notation: $w \propto \pi$) if $\pi\backslash w \neq \mathbf{0}$.

Example (Deriving the remaining protocol)

$$(a + (b \cdot c))^*\backslash b = (a\backslash b + (b \cdot c)\backslash b) \cdot (a + b \cdot c)^* =$$
$$(\mathbf{0} + (\mathbf{1} \cdot c)) \cdot (a + b \cdot c)^* = c \cdot (a + (b \cdot c))^*$$

The semantics

We interpret $\texttt{PDL}^!$ formulas on Kripke models $\mathcal{M} = (S, \rightarrow, V)$:

The semantics

We interpret $\mathtt{PDL}^!$ formulas on Kripke models $\mathcal{M} = (S, \to, V)$:

$$\mathcal{M}, s \vDash \phi \quad \Leftrightarrow \quad \mathcal{M}, s \vDash_{\boldsymbol{\Sigma}^*} \phi$$

The semantics

We interpret $\mathtt{PDL}^!$ formulas on Kripke models $\mathcal{M} = (S, \to, V)$:

$$\mathcal{M}, s \vDash \phi \quad \Leftrightarrow \quad \mathcal{M}, s \vDash_{\mathbf{\Sigma}^*} \phi$$
$$\mathcal{M}, s \vDash_{\pi} p \quad \Leftrightarrow \quad p \in V(s)$$

The semantics

We interpret PDL$^{!}$ formulas on Kripke models $\mathcal{M} = (S, \rightarrow, V)$:

$$
\begin{array}{rcl}
\mathcal{M}, s \vDash \phi & \Leftrightarrow & \mathcal{M}, s \vDash_{\Sigma^*} \phi \\
\mathcal{M}, s \vDash_\pi p & \Leftrightarrow & p \in V(s) \\
\mathcal{M}, s \vDash_\pi \neg\phi & \Leftrightarrow & \mathcal{M}, s \nvDash_\pi \phi
\end{array}
$$

The semantics

We interpret $\mathtt{PDL}^!$ formulas on Kripke models $\mathcal{M} = (S, \rightarrow, V)$:

$$
\begin{array}{rcl}
\mathcal{M}, s \vDash \phi & \Leftrightarrow & \mathcal{M}, s \vDash_{\mathbf{\Sigma}^*} \phi \\
\mathcal{M}, s \vDash_\pi p & \Leftrightarrow & p \in V(s) \\
\mathcal{M}, s \vDash_\pi \neg\phi & \Leftrightarrow & \mathcal{M}, s \nvDash_\pi \phi \\
\mathcal{M}, s \vDash_\pi \phi \wedge \psi & \Leftrightarrow & \mathcal{M}, s \vDash_\pi \phi \text{ and } \mathcal{M}, s \vDash_\pi \psi
\end{array}
$$

The semantics

We interpret $\texttt{PDL}^!$ formulas on Kripke models $\mathcal{M} = (S, \rightarrow, V)$:

$$
\begin{array}{rcl}
\mathcal{M}, s \vDash \phi & \Leftrightarrow & \mathcal{M}, s \vDash_{\mathbf{\Sigma}^*} \phi \\
\mathcal{M}, s \vDash_\pi p & \Leftrightarrow & p \in V(s) \\
\mathcal{M}, s \vDash_\pi \neg\phi & \Leftrightarrow & \mathcal{M}, s \nvDash_\pi \phi \\
\mathcal{M}, s \vDash_\pi \phi \wedge \psi & \Leftrightarrow & \mathcal{M}, s \vDash_\pi \phi \text{ and } \mathcal{M}, s \vDash_\pi \psi \\
\mathcal{M}, s \vDash_\pi [\pi']\phi & \Leftrightarrow & \text{for all } (w, s') : \text{ if } w \in \mathcal{L}(\pi'), w \propto \pi, s \xrightarrow{w} s' \\
 & & \text{then } \mathcal{M}, s' \vDash_{\pi \backslash w} \phi
\end{array}
$$

The semantics

We interpret PDL$^!$ formulas on Kripke models $\mathcal{M} = (S, \to, V)$:

$$
\begin{array}{rcl}
\mathcal{M}, s \vDash \phi & \Leftrightarrow & \mathcal{M}, s \vDash_{\mathbf{\Sigma}^*} \phi \\
\mathcal{M}, s \vDash_\pi p & \Leftrightarrow & p \in V(s) \\
\mathcal{M}, s \vDash_\pi \neg\phi & \Leftrightarrow & \mathcal{M}, s \nvDash_\pi \phi \\
\mathcal{M}, s \vDash_\pi \phi \wedge \psi & \Leftrightarrow & \mathcal{M}, s \vDash_\pi \phi \text{ and } \mathcal{M}, s \vDash_\pi \psi \\
\mathcal{M}, s \vDash_\pi [\pi']\phi & \Leftrightarrow & \text{for all } (w, s') : \text{ if } w \in \mathcal{L}(\pi'), w \propto \pi, s \xrightarrow{w} s' \\
 & & \text{then } \mathcal{M}, s' \vDash_{\pi \backslash w} \phi \\
\mathcal{M}, s \vDash_\pi [!\pi']\phi & \Leftrightarrow & \mathcal{M}, s \vDash \langle \pi' \rangle \top \implies \mathcal{M}, s \vDash_{\pi'} \phi
\end{array}
$$

where the mode $\mathbf{\Sigma}^*$ stands for the *universal protocol* $(a_0 + a_1 + \cdots + a_n)^*$ if $\mathbf{\Sigma} = \{a_0, a_1, \ldots, a_n\}$.

The semantics

$\mathcal{M}, s \models_{\pi} [\pi']\phi$	$\Leftrightarrow$	for all (w, s') : if $w \in \mathcal{L}(\pi'), w \propto \pi, s \xrightarrow{w} s'$ then $\mathcal{M}, s' \models_{\pi \backslash w} \phi$
$\mathcal{M}, s \models_{\pi} [!\pi']\phi$	$\Leftrightarrow$	$\mathcal{M}, s \models \langle \pi' \rangle \top \Longrightarrow \mathcal{M}, s \models_{\pi'} \phi$

Example

The semantics

$$\mathcal{M}, s \vDash_\pi [\pi']\phi \quad\Leftrightarrow\quad \text{for all } (w, s') : \text{ if } w \in \mathcal{L}(\pi'), w \propto \pi, s \xrightarrow{w} s' \text{ then } \mathcal{M}, s' \vDash_{\pi \backslash w} \phi$$

$$\mathcal{M}, s \vDash_\pi [!\pi']\phi \quad\Leftrightarrow\quad \mathcal{M}, s \vDash \langle \pi' \rangle \top \implies \mathcal{M}, s \vDash_{\pi'} \phi$$

Example

Consider the following model $\mathcal{M}$:

$s \vDash [!(a{\cdot}c+b{\cdot}d)][a+b](\neg\langle d\rangle\top \wedge \langle c\rangle\top \wedge [!(c+d)]\langle d\rangle\top)$

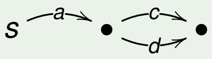
The semantics

$$\mathcal{M}, s \models_\pi [\pi']\phi \quad \Leftrightarrow \quad \text{for all } (w, s') : \text{ if } w \in \mathcal{L}(\pi'), w \propto \pi, s \xrightarrow{w} s' \text{ then } \mathcal{M}, s' \models_{\pi \backslash w} \phi$$

$$\mathcal{M}, s \models_\pi [!\pi']\phi \quad \Leftrightarrow \quad \mathcal{M}, s \models \langle \pi' \rangle \top \implies \mathcal{M}, s \models_{\pi'} \phi$$

Example

Consider the following model $\mathcal{M}$:



$s \models_{a \cdot c + b \cdot d} [a + b](\neg \langle d \rangle \top \wedge \langle c \rangle \top \wedge [!(c + d)]\langle d \rangle \top)$

The semantics

$$\mathcal{M}, s \vDash_\pi [\pi']\phi \quad \Leftrightarrow \quad \text{for all } (w, s') : \text{ if } w \in \mathcal{L}(\pi'), w \propto \pi, s \xrightarrow{w} s' \text{ then } \mathcal{M}, s' \vDash_{\pi \backslash w} \phi$$

$$\mathcal{M}, s \vDash_\pi [!\pi']\phi \quad \Leftrightarrow \quad \mathcal{M}, s \vDash \langle \pi' \rangle \top \implies \mathcal{M}, s \vDash_{\pi'} \phi$$

Example

Consider the following model $\mathcal{M}$:

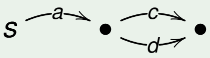
$s \vDash_c (\neg\langle d \rangle \top \wedge \langle c \rangle \top \wedge [!(c + d)]\langle d \rangle \top)$

The semantics

$$
\begin{array}{lcl}
\mathcal{M}, s \models_\pi [\pi']\phi & \Leftrightarrow & \text{for all } (w, s') : \text{ if } w \in \mathcal{L}(\pi'), w \propto \pi, s \xrightarrow{w} s' \\
 & & \text{then } \mathcal{M}, s' \models_{\pi \backslash w} \phi \\
\mathcal{M}, s \models_\pi [!\pi']\phi & \Leftrightarrow & \mathcal{M}, s \models \langle \pi' \rangle \top \implies \mathcal{M}, s \models_{\pi'} \phi
\end{array}
$$

Example

Consider the following model $\mathcal{M}$:



$s \models_{c+d} \langle d \rangle \top$

Expressivity

Theorem

$\mathtt{PDL}^{!}$ is equally expressive as test-free $\mathtt{PDL}$.

Expressivity

$$\mathcal{M}, s \vDash_\pi [\pi']\phi \quad \Leftrightarrow \quad \text{for all } (w, s') : \text{ if } w \in \mathcal{L}(\pi'), w \propto \pi, s \xrightarrow{w} s' \text{ then } \mathcal{M}, s' \vDash_{\pi \backslash w} \phi$$

Proof.

By a translation:

$$\begin{array}{rcl}
t(\phi) & = & t_{\Sigma^*}(\phi) \\
t_\pi(p) & = & p \\
t_\pi(\neg\phi) & = & \neg t_\pi(\phi) \\
t_\pi(\phi_1 \wedge \phi_2) & = & t_\pi(\phi_1) \wedge t_\pi(\phi_2) \\
t_\pi([\pi']\phi) & = & \bigwedge_{i=0}^{k}([\theta_i] t_{\pi \backslash \pi_i}(\phi)) \\
t_\pi([!\pi']\phi) & = & \langle \pi' \rangle \top \rightarrow t_{\pi'}(\phi)
\end{array}$$

The idea behind $t_\pi([\pi']\phi)$: partition $\{w \mid w \propto \pi \text{ and } w \in \mathcal{L}(\pi')\}$ by their consequences. □

Other variations

Introducing $[!\pi(x)]$

$$\mathcal{M}, s \vDash_\pi [!\pi'(x)]\phi \quad \Leftrightarrow \quad (\mathcal{M}, s \vDash \langle \pi'(\pi) \rangle \top \implies \mathcal{M}, s \vDash_{\pi'(\pi)} \phi)$$

We can then concatenate, add, insert and repeat protocols by announcing $x \cdot \pi'$, $x + \pi'$, $\pi' + x$, and x^* respectively.

Other variations

Introducing $[!\pi(x)]$

$$\mathcal{M}, s \vDash_\pi [!\pi'(x)]\phi \quad \Leftrightarrow \quad (\mathcal{M}, s \vDash \langle \pi'(\pi) \rangle \top \implies \mathcal{M}, s \vDash_{\pi'(\pi)} \phi)$$

We can then concatenate, add, insert and repeat protocols by announcing $x \cdot \pi'$, $x + \pi'$, $\pi' + x$, and x^* respectively.

Refinement operator $[!(a/\pi')]$

$$\mathcal{M}, s \vDash_\pi [!(a/\pi')]\phi \quad \Leftrightarrow \quad \mathcal{M}, s \vDash \langle \pi[a/\pi'] \rangle \top \implies \mathcal{M}, s \vDash_{\pi[a/\pi']} \phi$$

Public event logic

Public event logic

The language $\mathtt{PDL}^{!?_b}$

$$\begin{array}{lll} \phi & ::= & \top \mid p \mid \neg\phi \mid \phi \wedge \phi \mid [\pi']\phi \mid [!\pi]\phi \mid K_i\phi \\ \pi & ::= & ?\phi_b \mid a \mid \pi \cdot \pi \mid \pi + \pi \mid \pi^* \end{array}$$

π is test-free.

Public event logic

The language $\mathtt{PDL}^{!?_b}$

$$\phi ::= \top \mid p \mid \neg\phi \mid \phi \wedge \phi \mid [\pi']\phi \mid [!\pi]\phi \mid K_i\phi$$
$$\pi ::= ?\phi_b \mid a \mid \pi \cdot \pi \mid \pi + \pi \mid \pi^*$$

π is test-free. A *uniform guarded string* over finite sets **P** and **Σ** is a sequence $\rho a_1 \rho a_2 \rho \ldots \rho a_n \rho$ for some $\rho \subseteq \mathbf{P}$ [Koz01].

Public event logic

The language $\mathtt{PDL}^{!?_b}$

$$\begin{array}{lll} \phi & ::= & \top \mid p \mid \neg\phi \mid \phi \wedge \phi \mid [\pi']\phi \mid [!\pi]\phi \mid K_i\phi \\ \pi & ::= & ?\phi_b \mid a \mid \pi \cdot \pi \mid \pi + \pi \mid \pi^* \end{array}$$

π is test-free. A *uniform guarded string* over finite sets **P** and **Σ** is a sequence $\rho a_1 \rho a_2 \rho \ldots \rho a_n \rho$ for some $\rho \subseteq$ **P** [Koz01].

Public event logic

The language $\texttt{PDL}^{!?_b}$

$$\begin{array}{lll} \phi & ::= & \top \mid p \mid \neg\phi \mid \phi \wedge \phi \mid [\pi']\phi \mid [!\pi]\phi \mid K_i\phi \\ \pi & ::= & ?\phi_b \mid a \mid \pi \cdot \pi \mid \pi + \pi \mid \pi^* \end{array}$$

π is test-free. A *uniform guarded string* over finite sets **P** and **Σ** is a sequence $\rho a_1 \rho a_2 \rho \ldots \rho a_n \rho$ for some $\rho \subseteq \mathbf{P}$ [Koz01].

Example

If $\mathbf{P} = \{p\}$ then $\mathcal{L}_g(?p \cdot a \cdot b + ?\neg p \cdot a \cdot c) = \{\{p\}a\{p\}b\{p\}, \emptyset a \emptyset c \emptyset\}$

Public event logic

The language $\texttt{PDL}^{!?_b}$

$$\begin{array}{lcl} \phi & ::= & \top \mid p \mid \neg\phi \mid \phi \wedge \phi \mid [\pi']\phi \mid [!\pi]\phi \mid K_i\phi \\ \pi & ::= & ?\phi_b \mid a \mid \pi \cdot \pi \mid \pi + \pi \mid \pi^* \end{array}$$

π is test-free. A *uniform guarded string* over finite sets **P** and **Σ** is a sequence $\rho a_1 \rho a_2 \rho \ldots \rho a_n \rho$ for some $\rho \subseteq \mathbf{P}$ [Koz01].

Example

If $\mathbf{P} = \{p\}$ then $\mathcal{L}_g(?p \cdot a \cdot b + ?\neg p \cdot a \cdot c) = \{\{p\}a\{p\}b\{p\}, \emptyset a \emptyset c \emptyset\}$

Public event logic

The language $\texttt{PDL}^{!?_b}$

$$\begin{array}{lll}\phi & ::= & \top \mid p \mid \neg\phi \mid \phi \wedge \phi \mid [\pi']\phi \mid [!\pi]\phi \mid K_i\phi \\ \pi & ::= & ?\phi_b \mid a \mid \pi \cdot \pi \mid \pi + \pi \mid \pi^*\end{array}$$

π is test-free. A *uniform guarded string* over finite sets **P** and **Σ** is a sequence $\rho a_1 \rho a_2 \rho \ldots \rho a_n \rho$ for some $\rho \subseteq \mathbf{P}$ [Koz01].

Example

If **P** $= \{p\}$ then $\mathcal{L}_g(?p \cdot a \cdot b + ?\neg p \cdot a \cdot c) = \{\{p\}a\{p\}b\{p\}, \emptyset a \emptyset c \emptyset\}$

Example

$\mathcal{L}_g((?p \cdot a \cdot b + ?\neg p \cdot a \cdot c + b) \backslash a) = \{\{p\}b\{p\}, \emptyset c \emptyset\} = \mathcal{L}_g(?p \cdot b + ?\neg p \cdot c)$.

The semantics

Now we interpret $\mathtt{PDL}^{!?_b}$ on the $S5$ models $(S, \{\sim_i\}_{i \in \mathbf{I}}, V)$ as follows:

$$
\begin{array}{rcl}
\mathcal{M}, s \vDash \phi & \Leftrightarrow & \mathcal{M}, s \vDash_{\mathbf{\Sigma}^*} \phi \\
\mathcal{M}, s \vDash_\pi p & \Leftrightarrow & p \in V(s) \\
\mathcal{M}, s \vDash_\pi \neg\phi & \Leftrightarrow & \mathcal{M}, s \nvDash_\pi \phi \\
\mathcal{M}, s \vDash_\pi \phi \wedge \phi' & \Leftrightarrow & \mathcal{M}, s \vDash_\pi \phi \text{ and } \mathcal{M}, s \vDash_\pi \phi' \\
\mathcal{M}, s \vDash K_i\phi & \Leftrightarrow & \text{for all } t\text{: } s \sim_i t \implies \mathcal{M}, t \vDash \phi \\
\mathcal{M}, s \vDash_\pi [\pi']\phi & \Leftrightarrow & \text{for all } w \in \mathcal{L}(\pi') : \text{ if } \mathcal{M}, s \vDash \phi^w_\pi \\
 & & \text{ then } \mathcal{M}|_{\phi^w_\pi}, s \vDash_{\pi \backslash w} \phi \\
\mathcal{M}, s \vDash_\pi [!\pi']\phi & \Leftrightarrow & \text{ if } (\exists w : w = \rho v \in \mathcal{L}_g(\pi') \text{ and } V(s) = \rho) \\
 & & \text{ then } \mathcal{M}, s \vDash_{\pi'} \phi
\end{array}
$$

The semantics

Now we interpret $\mathtt{PDL}^{!?_b}$ on the $S5$ models $(S, \{\sim_i\}_{i\in\mathbf{I}}, V)$ as follows:

$$
\begin{array}{rcl}
\mathcal{M}, s \vDash \phi & \Leftrightarrow & \mathcal{M}, s \vDash_{\boldsymbol{\Sigma}^*} \phi \\
\mathcal{M}, s \vDash_\pi p & \Leftrightarrow & p \in V(s) \\
\mathcal{M}, s \vDash_\pi \neg\phi & \Leftrightarrow & \mathcal{M}, s \nvDash_\pi \phi \\
\mathcal{M}, s \vDash_\pi \phi \wedge \phi' & \Leftrightarrow & \mathcal{M}, s \vDash_\pi \phi \text{ and } \mathcal{M}, s \vDash_\pi \phi' \\
\mathcal{M}, s \vDash K_i\phi & \Leftrightarrow & \text{for all } t\text{: } s \sim_i t \implies \mathcal{M}, t \vDash \phi \\
\mathcal{M}, s \vDash_\pi [\pi']\phi & \Leftrightarrow & \text{for all } w \in \mathcal{L}(\pi') : \text{ if } \mathcal{M}, s \vDash \phi^w_\pi \\
 & & \text{then } \mathcal{M}|_{\phi^w_\pi}, s \vDash_{\pi\backslash w} \phi \\
\mathcal{M}, s \vDash_\pi [!\pi']\phi & \Leftrightarrow & \text{if } (\exists w : w = \rho v \in \mathcal{L}_g(\pi') \text{ and } V(s) = \rho) \\
 & & \text{then } \mathcal{M}, s \vDash_{\pi'} \phi
\end{array}
$$

where:

$\phi^w_\pi = \bigvee\{\phi_\rho \mid v = \rho a_1 \rho a_2 \rho \cdots \rho a_k \rho, \mathcal{L}_p(v) = w, v \propto_g \pi\}$.

The semantics

Now we interpret $\mathtt{PDL}^{!?b}$ on the $S5$ models $(S, \{\sim_i\}_{i\in\mathbf{I}}, V)$ as follows:

$$
\begin{array}{rcl}
\mathcal{M}, s \vDash \phi & \Leftrightarrow & \mathcal{M}, s \vDash_{\mathbf{\Sigma}^*} \phi \\
\mathcal{M}, s \vDash_\pi p & \Leftrightarrow & p \in V(s) \\
\mathcal{M}, s \vDash_\pi \neg\phi & \Leftrightarrow & \mathcal{M}, s \nvDash_\pi \phi \\
\mathcal{M}, s \vDash_\pi \phi \wedge \phi' & \Leftrightarrow & \mathcal{M}, s \vDash_\pi \phi \text{ and } \mathcal{M}, s \vDash_\pi \phi' \\
\mathcal{M}, s \vDash K_i\phi & \Leftrightarrow & \text{for all } t\text{: } s \sim_i t \implies \mathcal{M}, t \vDash \phi \\
\mathcal{M}, s \vDash_\pi [\pi']\phi & \Leftrightarrow & \text{for all } w \in \mathcal{L}(\pi') : \text{ if } \mathcal{M}, s \vDash \phi^w_\pi \\
& & \text{then } \mathcal{M}|_{\phi^w_\pi}, s \vDash_{\pi\backslash w} \phi \\
\mathcal{M}, s \vDash_\pi [!\pi']\phi & \Leftrightarrow & \text{if } (\exists w : w = \rho v \in \mathcal{L}_g(\pi') \text{ and } V(s) = \rho) \\
& & \text{then } \mathcal{M}, s \vDash_{\pi'} \phi
\end{array}
$$

where:
$\phi^w_\pi = \bigvee\{\phi_\rho \mid v = \rho a_1 \rho a_2 \rho \cdots \rho a_k \rho, \mathcal{L}_p(v) = w, v \propto_g \pi\}$.For example, let $\pi = ?p \cdot a \cdot b + ?\neg p \cdot a \cdot c$, $w = a$, then $\phi^w_a = p \vee \neg p$.

The semantics

Example

Consider the following model $\mathcal{M}$:

$$s : p \overset{1}{—} t : \neg p$$

The semantics

Example

Consider the following model $\mathcal{M}$:

$$s : p \overset{1}{\text{——}} t : \neg p$$

$$\mathcal{M}, s \vDash [!(?p \cdot a \cdot b + ?\neg p \cdot a \cdot c)][a](\neg K_1 p \wedge [b]K_1 p)$$

The semantics

Example

Consider the following model $\mathcal{M}$:

$$s : p \overset{1}{—} t : \neg p$$

$\mathcal{M}, s \models [!(?p \cdot a \cdot b + ?\neg p \cdot a \cdot c)][a](\neg K_1 p \wedge [b]K_1 p)$
$\iff \mathcal{M}, s \models_{?p \cdot a \cdot b + ?\neg p \cdot a \cdot c} [a](\neg K_1 p \wedge [b]K_1 p)$

The semantics

Example

Consider the following model $\mathcal{M}$:

$$s : p \overset{1}{\text{——}} t : \neg p$$

$$\mathcal{M}, s \vDash [!(?p \cdot a \cdot b + ?\neg p \cdot a \cdot c)][a](\neg K_1 p \wedge [b]K_1 p)$$
$$\iff \mathcal{M}, s \vDash_{?p \cdot a \cdot b + ?\neg p \cdot a \cdot c} [a](\neg K_1 p \wedge [b]K_1 p)$$
$$\iff \mathcal{M}|_{p \vee \neg p}, s \vDash_{?p \cdot b + ?\neg p \cdot c} (\neg K_1 p \wedge [b]K_1 p)$$

The semantics

Example

Consider the following model $\mathcal{M}$:

$$s : p \text{ —}_1\text{— } t : \neg p$$

$$\mathcal{M}, s \vDash [!(?p \cdot a \cdot b + ?\neg p \cdot a \cdot c)][a](\neg K_1 p \wedge [b]K_1 p)$$
$$\iff \mathcal{M}, s \vDash_{?p \cdot a \cdot b + ?\neg p \cdot a \cdot c} [a](\neg K_1 p \wedge [b]K_1 p)$$
$$\iff \mathcal{M}|_{p \vee \neg p}, s \vDash_{?p \cdot b + ?\neg p \cdot c} (\neg K_1 p \wedge [b]K_1 p)$$
$$\iff \mathcal{M}, s \vDash_{?p \cdot b + ?\neg p \cdot c} \neg K_1 p \text{ and } \mathcal{M}, s \vDash_{?p \cdot b + ?\neg p \cdot c} [b]K_1 p$$

The semantics

Example

Consider the following model $\mathcal{M}$:

$$s : p \overset{1}{—} t : \neg p$$

$$\mathcal{M}, s \vDash [!(?p \cdot a \cdot b + ?\neg p \cdot a \cdot c)][a](\neg K_1 p \wedge [b]K_1 p)$$
$$\iff \mathcal{M}, s \vDash_{?p\cdot a\cdot b+?\neg p\cdot a\cdot c} [a](\neg K_1 p \wedge [b]K_1 p)$$
$$\iff \mathcal{M}|_{p\vee\neg p}, s \vDash_{?p\cdot b+?\neg p\cdot c} (\neg K_1 p \wedge [b]K_1 p)$$
$$\iff \mathcal{M}, s \vDash_{?p\cdot b+?\neg p\cdot c} \neg K_1 p \text{ and } \mathcal{M}, s \vDash_{?p\cdot b+?\neg p\cdot c} [b]K_1 p$$
$$\iff \mathcal{M}, s \vDash \neg K_1 p \text{ and } \mathcal{M}|_p, s \vDash_{?p} K_1 p$$

The semantics

Example

Consider the following model $\mathcal{M}$:

$$s : p \overset{1}{—} t : \neg p$$

$\mathcal{M}, s \vDash [!(?p \cdot a \cdot b + ?\neg p \cdot a \cdot c)][a](\neg K_1 p \wedge [b]K_1 p)$

$\iff \mathcal{M}, s \vDash_{?p\cdot a\cdot b+?\neg p\cdot a\cdot c} [a](\neg K_1 p \wedge [b]K_1 p)$

$\iff \mathcal{M}|_{p\vee\neg p}, s \vDash_{?p\cdot b+?\neg p\cdot c} (\neg K_1 p \wedge [b]K_1 p)$

$\iff \mathcal{M}, s \vDash_{?p\cdot b+?\neg p\cdot c} \neg K_1 p$ and $\mathcal{M}, s \vDash_{?p\cdot b+?\neg p\cdot c} [b]K_1 p$

$\iff \mathcal{M}, s \vDash \neg K_1 p$ and $\mathcal{M}|_p, s \vDash_{?p} K_1 p$

$\iff \mathcal{M}, s \vDash \neg K_1 p$ and $\mathcal{M}, s \vDash_{\mathrm{PAL}} [!p]K_1 p$

The semantics

Example

Consider the following model $\mathcal{M}$:

$$s : p \text{ —1— } t : \neg p$$

$\mathcal{M}, s \vDash [!(?p \cdot a \cdot b + ?\neg p \cdot a \cdot c)][a](\neg K_1 p \wedge [b]K_1 p)$

$\iff \mathcal{M}, s \vDash_{?p \cdot a \cdot b + ?\neg p \cdot a \cdot c} [a](\neg K_1 p \wedge [b]K_1 p)$

$\iff \mathcal{M}|_{p \vee \neg p}, s \vDash_{?p \cdot b + ?\neg p \cdot c} (\neg K_1 p \wedge [b]K_1 p)$

$\iff \mathcal{M}, s \vDash_{?p \cdot b + ?\neg p \cdot c} \neg K_1 p$ and $\mathcal{M}, s \vDash_{?p \cdot b + ?\neg p \cdot c} [b]K_1 p$

$\iff \mathcal{M}, s \vDash \neg K_1 p$ and $\mathcal{M}|_p, s \vDash_{?p} K_1 p$

$\iff \mathcal{M}, s \vDash \neg K_1 p$ and $\mathcal{M}, s \vDash_{\text{PAL}} [!p]K_1 p$

Consider the Häagen-Dazs protocol: $\pi_{H\text{-}D} = ?p_{love} \cdot a_{buy}$, $[!\pi_{H\text{-}D}][a_{buy}]K_i p_{love}$ is valid. However, buying an ice cream without the announcement $!\pi_{H\text{-}D}$ does not mean anything: $[a_{buy}]K_i p_{love}$ is not valid.

Expressivity

Theorem

$\mathtt{PDL}^{!?_b}$ is equally expressive as $\mathtt{PAL}$ on S5 models.

Expressivity

Theorem

$\mathtt{PDL}^{!?_b}$ is equally expressive as $\mathtt{PAL}$ on S5 models.

Proof.

We can define the following translation from $\mathtt{PDL}^{!?_b}$ to $\mathtt{PAL}$:

$$
\begin{array}{rcl}
t(\phi) & = & t_{\boldsymbol{\Sigma}^*}(\phi) \\
t_\pi(p) & = & p \\
t_\pi(\neg\phi) & = & \neg t_\pi(\phi) \\
t_\pi(\phi_1 \wedge \phi_2) & = & t_\pi(\phi_1) \wedge t_\pi(\phi_2) \\
t_\pi(K_i\phi) & = & K_i t_\pi(\phi) \\
t_\pi([\pi']\phi) & = & \bigwedge\{[!\psi_j]t_{\theta_j}(\phi) \mid \mathcal{L}(\pi_j) \cap \mathcal{L}(\pi') \neq \emptyset\} \\
t_\pi([!\pi']\phi) & = & \chi_{\pi'} \to t_{\pi'}(\phi)
\end{array}
$$

□

Expressivity

Theorem

$\mathtt{PDL}^{!?_b}$ is equally expressive as $\mathtt{PAL}$ on S5 models.

Note that $[!p](K_i p \wedge [!q]q)$ can be reinterpreted in $\mathtt{PDL}^{!?_b}$ as

$$[!(?p \cdot a + ?q \cdot b)^*][a](K_i p \wedge [b]q).$$

Thus we can separate actions from their meanings.

Composing models: Joint work with van Eijck and Sietsma

Composing models: Joint work with van Eijck and Sietsma

Aristotle said...

Love is composed of a single soul inhabiting two bodies.

Composing models: Joint work with van Eijck and Sietsma

Aristotle said...

Love is composed of a single soul inhabiting two bodies.

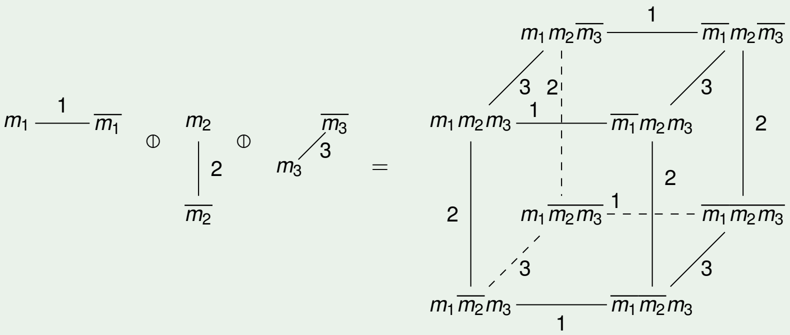
Definition (Merging Composition of "Partial" Kripke Models)

Given two models with the same set of agents $\mathbf{I}$: $\mathcal{M} = (S, \mathbf{P}, \mathbf{I}, \sim, V)$ and $\mathcal{N} = (T, \mathbf{P}', \mathbf{I}, \sim', V')$, the *merging composition* $\mathcal{M} \oplus \mathcal{N}$ is given by $(S'', \mathbf{P} \cup \mathbf{P}', \mathbf{I}, \sim'', V'')$, where:

- $S'' = \{(s, t) \mid s \in S, t \in T, V(s) \cap \mathbf{P}' = V'(t) \cap \mathbf{P}\}$,
- $(s, s') \sim_i'' (t, t')$ iff $s \sim_i t$ and $s' \sim_i' t'$,
- $V''(s, t) = V(s) \cup V'(t)$.

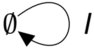
Composing models

Example (Composing Muddy Children)



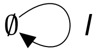
Composing models

We define the *unit* model $\mathcal{E}$ as the model $(\{s\}, \emptyset, \mathbf{I}, \sim, V)$ where $V(s) = \emptyset$ and $\sim_i = \{(s, s)\}$ for any i. In a picture:



Composing models

We define the *unit* model $\mathcal{E}$ as the model $(\{s\}, \emptyset, \mathbf{I}, \sim, V)$ where $V(s) = \emptyset$ and $\sim_i = \{(s,s)\}$ for any i. In a picture:



Theorem

Kripke models with the same set of agents form a commutative monoid under the $\oplus$ operation, with total bisimilarity as the appropriate equality notion. In particular, we have:

$$
\begin{array}{rcl}
\mathcal{E} \oplus \mathcal{M} & \underline{\leftrightarrow} & \mathcal{M} \\
\mathcal{M} \oplus \mathcal{E} & \underline{\leftrightarrow} & \mathcal{M} \\
\mathcal{M} \oplus (\mathcal{N} \oplus \mathcal{K}) & \underline{\leftrightarrow} & (\mathcal{M} \oplus \mathcal{N}) \oplus \mathcal{K} \\
\mathcal{M} \oplus \mathcal{N} & \underline{\leftrightarrow} & \mathcal{N} \oplus \mathcal{M}
\end{array}
$$

Composing models

The commutative monoid yields the algebraic preordering $\leq$ on the class of Kripke models with different vocabularies:

$$\mathcal{M} \leq \mathcal{N} \text{ iff there is a } \mathcal{K} \text{ with } \mathcal{M} \oplus \mathcal{K} \leftrightarrows \mathcal{N}.$$

Composing models

The commutative monoid yields the algebraic preordering $\leq$ on the class of Kripke models with different vocabularies:

$$\mathcal{M} \leq \mathcal{N} \text{ iff there is a } \mathcal{K} \text{ with } \mathcal{M} \oplus \mathcal{K} \leftrightarrows \mathcal{N}.$$

Given two models $\mathcal{M}$ and $\mathcal{N}$ such that $\mathbf{P}_{\mathcal{M}} \subseteq \mathbf{P}_{\mathcal{N}}$, a left-simulation between $\mathcal{M}$ and $\mathcal{N}$ is a relation $R \subseteq S_{\mathcal{M}} \times S_{\mathcal{N}}$ such that sRt implies that the following hold:

Restricted Invariance $V_{\mathcal{M}}(s) = V_{\mathcal{N}}(t) \cap \mathbf{P}_{\mathcal{M}}$;

Zag If for some $i \in \mathbf{I}$ there is a $t' \in S_{\mathcal{N}}$ with $t \xrightarrow{i} t'$ then there is a $s' \in S_{\mathcal{M}}$ with $s \xrightarrow{i} s'$ and $s'Rt'$.

Composing models

Theorem

For any models $\mathcal{M}, \mathcal{N}$ with arbitrary vocabularies:

$$\mathcal{M} \leq \mathcal{N} \implies \mathcal{M} \leftleftharpoons \mathcal{N}$$

Composing models

Theorem

For any models $\mathcal{M}, \mathcal{N}$ with arbitrary vocabularies:

$$\mathcal{M} \leq \mathcal{N} \implies \mathcal{M} \leftleftharpoons \mathcal{N}$$

Theorem

Let $\mathcal{M}$ be a propositionally differentiated model. Then

$$\mathcal{M} \leq \mathcal{N} \iff \mathcal{M} \leftleftharpoons \mathcal{N}$$

Composing models

Theorem

If $\mathcal{M}, s \leq \mathcal{N}, t$ then all formulas ϕ in the diamond fragment of $\mathit{PDL}_{\mathbf{P}_{\mathcal{M}}, \mathbf{I}}$ are preserved from right to left under left simulation: if $\mathcal{N}, t \models \phi$ then $\mathcal{M}, s \models \phi$. Equivalently, the box fragment of $\mathit{PDL}_{\mathbf{P}_{\mathcal{M}}, \mathbf{I}}$ is preserved from left to right under left simulation.

Composing models

Theorem

If $\mathcal{M}, s \leq \mathcal{N}, t$ then all formulas ϕ in the diamond fragment of $PDL_{\mathbf{P}_{\mathcal{M}}, \mathbf{I}}$ are preserved from right to left under left simulation: if $\mathcal{N}, t \vDash \phi$ then $\mathcal{M}, s \vDash \phi$. Equivalently, the box fragment of $PDL_{\mathbf{P}_{\mathcal{M}}, \mathbf{I}}$ is preserved from left to right under left simulation.

Theorem

If a pointed model $(\mathcal{M}, s)$ is decomposable into models $(\mathcal{M}_0, s_0), \ldots, (\mathcal{M}_n, s_n)$ with disjoint vocabularies $\mathbf{P}_0, \mathbf{P}_1, \ldots, \mathbf{P}_n$, then for any i: $\mathcal{M}_i, s_i \leftrightarroweq_{\mathbf{P}_i} \mathcal{M}, s$. Therefore for any ϕ in $PDL_{\mathbf{P}_i, \mathbf{I}} : \mathcal{M}_i, s_i \vDash \phi \iff \mathcal{M}, s \vDash \phi$.

Composing models

We say $\mathcal{M}$ is *locally generated* if, for every agent i, there is a non-empty set of boolean formulas Φ_i (the set of local observables) based on $\mathbf{P}_{\mathcal{M}}$ such that:

$$\text{for all } s, s' \in S_{\mathcal{M}}, s \sim_i s' \text{ iff for all } \varphi \in \Phi_i, \mathcal{M}, s \models \varphi \Leftrightarrow \mathcal{M}, s' \models \varphi$$

Composing models

We say $\mathcal{M}$ is *locally generated* if, for every agent i, there is a non-empty set of boolean formulas Φ_i (the set of local observables) based on $\mathbf{P}_{\mathcal{M}}$ such that:

for all $s, s' \in S_{\mathcal{M}}, s \sim_i s'$ iff for all $\varphi \in \Phi_i, \mathcal{M}, s \models \varphi \Leftrightarrow \mathcal{M}, s' \models \varphi$

Theorem (Decomposition by agents)

Given a set of agent $\mathbf{I} = \{1, 2, \ldots n\}$. If $\mathcal{M} = (S, \mathbf{P}, \mathbf{I}, \sim, V)$ is locally generated w.r.t. $\Phi_1, \ldots, \Phi_n$, then there are models $\mathcal{M}_1, \ldots, \mathcal{M}_n$ and $\mathcal{M}_0$ such that:

- $\mathcal{M} \leftrightarrows (\mathcal{M}_0 \oplus \mathcal{M}_1 \oplus \cdots \oplus \mathcal{M}_n)$;
- $|S_{\mathcal{M}_j}| \leq |S|$ *and* $\mathcal{M}_i$ *is bisimulation contracted model;*
- $\mathbf{P}_{\mathcal{M}_j} = \{p \in \mathbf{P}_{\mathcal{M}} \mid p \text{ appears in } \Phi_j\}$ *for* $j > 0$;

Composing models

Theorem (Decomposition by issues)

Given a set of agent $\mathbf{I} = \{1, 2, \ldots n\}$ and a set of proposition letters $\mathbf{P} = \{p_1, \ldots, p_k\}$, if $\mathcal{M} = (S, \mathbf{P}, \mathbf{I}, \sim, V)$ is locally generated by $\Phi_1, \ldots, \Phi_n$ such that Φ_i only contains atomic propositions (i.e., $\Phi_i \subseteq \mathbf{P}$), then there are models $\mathcal{M}_1, \ldots, \mathcal{M}_k$ and $\mathcal{M}_0$ such that:

- $\mathcal{M} \leftrightarrows (\mathcal{M}_0 \oplus \mathcal{M}_1 \oplus \cdots \oplus \mathcal{M}_k)$;
- $\mathbf{P}_{\mathcal{M}_j} = \{p_j\}$ *for* $j > 0$ *and* $\mathbf{P}_0 = \mathbf{P}$;
- $|S_{\mathcal{M}_j}| = 2$ *for* $j > 0$

Composing models

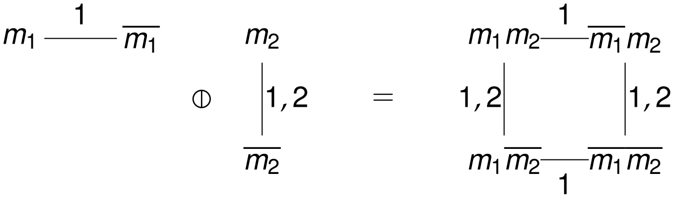
Definition (Model expansion)

Given $\mathcal{M}$ we define the *expansion* of $\mathcal{M}$ w.r.t. vocabulary $\mathbf{P}'$ as follows: $\mathcal{M} \triangleleft \mathbf{P}' = \mathcal{M} \oplus \mathcal{M}^{\mathbf{I}}_{\mathbf{P}'}$.

Composing models

Definition (Model expansion)

Given $\mathcal{M}$ we define the *expansion* of $\mathcal{M}$ w.r.t. vocabulary $\mathbf{P}'$ as follows: $\mathcal{M} \triangleleft \mathbf{P}' = \mathcal{M} \oplus \mathcal{M}^{\mathbf{I}}_{\mathbf{P}'}$.



Composing models

Definition (Extended Product Update)

Given a static model $\mathcal{M}$ and an event model $\mathcal{A}$ for the same set of agents $\mathbf{I}$. Let $X = \mathbf{P}_{\mathcal{A}} - \mathbf{P}_{\mathcal{M}}$. Then the extended product update $\mathcal{M} \circledcirc \mathcal{A}$ is the static model defined by $(\mathcal{M} \triangleleft X) \otimes \mathcal{A}$.

Composing models

Definition (Extended Product Update)

Given a static model $\mathcal{M}$ and an event model $\mathcal{A}$ for the same set of agents $\mathbf{I}$. Let $X = \mathbf{P}_{\mathcal{A}} - \mathbf{P}_{\mathcal{M}}$. Then the extended product update $\mathcal{M} \circledcirc \mathcal{A}$ is the static model defined by $(\mathcal{M} \triangleleft X) \otimes \mathcal{A}$.

Theorem

When $\mathcal{A}$ is propositionally differentiated:
$(\mathcal{M} \oplus \mathcal{N}) \circledcirc \mathcal{A} \leftrightarrows (\mathcal{M} \circledcirc \mathcal{A}) \oplus (\mathcal{N} \circledcirc \mathcal{A})$

Composing models

Definition (Extended Product Update)

Given a static model $\mathcal{M}$ and an event model $\mathcal{A}$ for the same set of agents **I**. Let $X = \mathbf{P}_{\mathcal{A}} - \mathbf{P}_{\mathcal{M}}$. Then the extended product update $\mathcal{M} \circledcirc \mathcal{A}$ is the static model defined by $(\mathcal{M} \triangleleft X) \otimes \mathcal{A}$.

Theorem

When $\mathcal{A}$ is propositionally differentiated:
$(\mathcal{M} \oplus \mathcal{N}) \circledcirc \mathcal{A} \leftrightarrows (\mathcal{M} \circledcirc \mathcal{A}) \oplus (\mathcal{N} \circledcirc \mathcal{A})$

Theorem

Let $\mathcal{A} \oplus \mathcal{B}$ be the composition of event models:
$\mathcal{M} \circledcirc (\mathcal{A} \oplus \mathcal{B}) \leftrightarrows (\mathcal{M} \circledcirc \mathcal{A}) \oplus (\mathcal{M} \circledcirc \mathcal{B})$.

Counting models: joint work with Siestma

Counting models: joint work with Siestma

In *Anna Karenina*:

All happy families are happy alike, all unhappy families are unhappy in their own way.

Counting models: joint work with Siestma

In *Anna Karenina*:

All happy families are happy alike, all unhappy families are unhappy in their own way.

My question is...

How many different kinds of unhappiness are there?

Counting models: joint work with Siestma

In *Anna Karenina*:

All happy families are happy alike, all unhappy families are unhappy in their own way.

In other words...

Given a finite set of *formulas*, how many *different* models are there?

Counting models: joint work with Siestma

In *Anna Karenina*:

All happy families are happy alike, all unhappy families are unhappy in their own way.

In other words...

Given a finite set of *formulas*, how many *different* models are there?

To be more precise, we consider:

- modal μ–calculus formulas
- bisimulation as the equivalence notion between models
- only *image-finite* models

Counting models

Modal μ-calculus [Koz83]

$$\phi \quad ::= \quad \top \mid \bot \mid X \mid p \mid \bar{p} \mid \phi \wedge \phi \mid \phi \vee \phi \mid \langle a \rangle \phi \mid [a]\phi \mid \mu X.\phi \mid \nu X.\phi$$

Very expressive: $\mu X.\Box X$ which expresses well-foundedness. It is shown in [vBI08] that `Mu` is closed under product update.

Counting models

Modal μ-calculus [Koz83]

$$\phi \quad ::= \quad \top \mid \bot \mid X \mid p \mid \bar{p} \mid \phi \wedge \phi \mid \phi \vee \phi \mid \langle a \rangle \phi \mid [a]\phi \mid \mu X.\phi \mid \nu X.\phi$$

Very expressive: $\mu X.\Box X$ which expresses well-foundedness. It is shown in [vBI08] that Mu is closed under product update.

However, we do not work with formulas directly but..

Counting models

Modal μ-calculus [Koz83]

$$\phi \quad ::= \quad \top \mid \bot \mid X \mid p \mid \bar{p} \mid \phi \wedge \phi \mid \phi \vee \phi \mid \langle a \rangle \phi \mid [a]\phi \mid \mu X.\phi \mid \nu X.\phi$$

Very expressive: $\mu X.\Box X$ which expresses well-foundedness. It is shown in [vBI08] that `Mu` is closed under product update.

However, we do not work with formulas directly but..

Definition (μ-automata [JW95, DN05])

A μ-automaton A on set of basic propositions **P** and set of basic actions **Σ** is a tuple: $A = (Q, B, q_0, \rightarrow_{OR}, \rightarrow_{BR}, L, \Omega)$.

Counting models

Modal μ-calculus [Koz83]

$$\phi \quad ::= \quad \top \mid \bot \mid X \mid p \mid \bar{p} \mid \phi \wedge \phi \mid \phi \vee \phi \mid \langle a \rangle \phi \mid [a]\phi \mid \mu X.\phi \mid \nu X.\phi$$

Very expressive: $\mu X.\Box X$ which expresses well-foundedness. It is shown in [vBI08] that `Mu` is closed under product update.

However, we do not work with formulas directly but..

Definition (μ-automata [JW95, DN05])

A μ-automaton A on set of basic propositions **P** and set of basic actions **Σ** is a tuple: $A = (Q, B, q_0, \rightarrow_{OR}, \rightarrow_{BR}, L, \Omega)$.

μ automata recognizes (infinite) trees. Let $\mathcal{L}(A)$ be the language of A, i.e., the set of trees which are accepted by A.

Counting models

Theorem ([JW95])

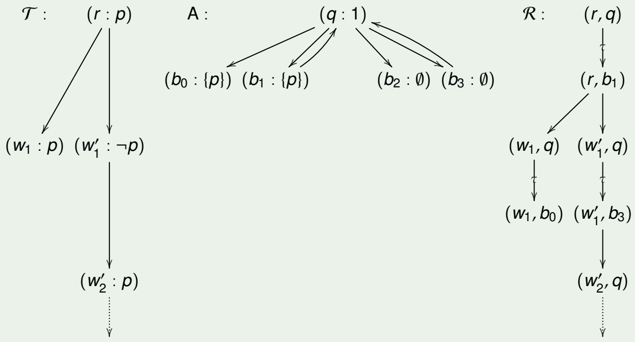
For each μ–automaton there is an equivalent $\mathtt{Mu}$–*formula. For each* $\mathtt{Mu}$–*formula there is an equivalent μ–automaton .*

Counting models

Theorem ([JW95])

For each μ–automaton there is an equivalent Mu*–formula. For each* Mu*–formula there is an equivalent μ–automaton .*

Example ($\mu X.\Box X$ and its μ-automata)



Counting models

Theorem

Let A be a μ-automaton. Then the following are equivalent:

1. $|\mathcal{L}(A)|_{/\underline{\leftrightarrow}} = 2^{\aleph_0}$,
2. $|\mathcal{L}(A)|_{/\underline{\leftrightarrow}} > \aleph_0$,
3. $\mathcal{L}(A)$ *contains a tree with infinite non-bisimilar subtrees (non-B-regular tree).*

This generalizes an earlier work by Niwiński [Niw91].

Counting models

Theorem

Let A be a μ-automaton. Then the following are equivalent:

1. $|\mathcal{L}(A)|_{/\underline{\leftrightarrow}} = 2^{\aleph_0}$,
2. $|\mathcal{L}(A)|_{/\underline{\leftrightarrow}} > \aleph_0$,
3. $\mathcal{L}(A)$ *contains a tree with infinite non-bisimilar subtrees (non-B-regular tree).*

This generalizes an earlier work by Niwiński [Niw91].

Proof.

Hard part: (3) $\Longrightarrow$ (1). The strategy is as follows:

1. Show every non-B-tree has an infinite ”non-bisimilar ” path.

Counting models

Theorem

Let A be a μ-automaton. Then the following are equivalent:

1. $|\mathcal{L}(A)|_{/\underline{\leftrightarrow}} = 2^{\aleph_0}$,
2. $|\mathcal{L}(A)|_{/\underline{\leftrightarrow}} > \aleph_0$,
3. $\mathcal{L}(A)$ *contains a tree with infinite non-bisimilar subtrees (non-B-regular tree).*

This generalizes an earlier work by Niwiński [Niw91].

Proof.

Hard part: (3) $\Longrightarrow$ (1). The strategy is as follows:

1. Show every non-B-tree has an infinite "non-bisimilar " path.
2. Based on the special path, "Pump" a non-B-regular tree in $\mathcal{L}$(A) tree into $2^{\aleph}_{0}$ non-bisimilar trees.

Counting models

Theorem

Let A be a μ-automaton. Then the following are equivalent:

1. $|\mathcal{L}(A)|_{/\underline{\leftrightarrow}} = 2^{\aleph_0}$,
2. $|\mathcal{L}(A)|_{/\underline{\leftrightarrow}} > \aleph_0$,
3. *$\mathcal{L}(A)$ contains a tree with infinite non-bisimilar subtrees (non-B-regular tree).*

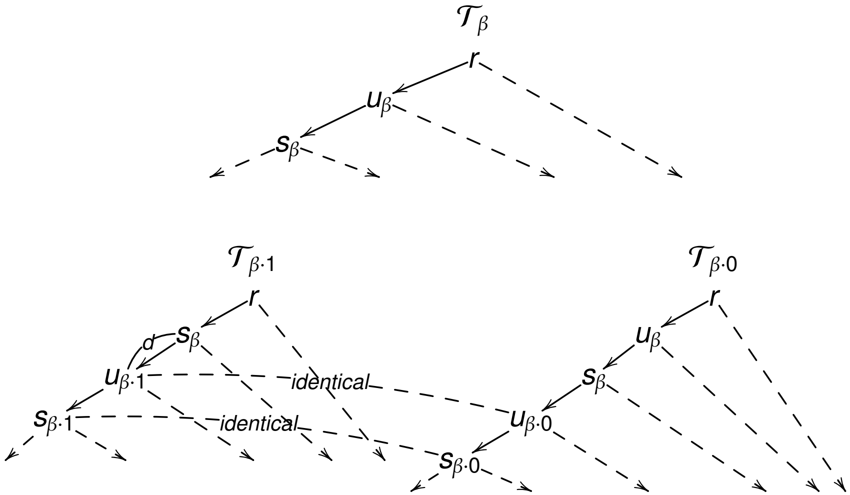
This generalizes an earlier work by Niwiński [Niw91].

Proof.

Hard part: (3) $\Longrightarrow$ (1). The strategy is as follows:

1. Show every non-B-tree has an infinite ”non-bisimilar ” path.
2. Based on the special path, “Pump” a non-B-regular tree in $\mathcal{L}$(A) tree into $2^{\aleph}_{0}$ non-bisimilar trees.
3. Show that all these tree are accepted by A.

Counting models



Counting models

Lemma

Given a μ–automaton A, if $\mathcal{L}(A)$ is countable up to bisimulation then $|alive(\mathcal{L}(A))|_{/\underline{\leftrightarrow}}$ is finite.

Counting models

Lemma

Given a μ–automaton A, if $\mathcal{L}(A)$ is countable up to bisimulation then $|alive(\mathcal{L}(A))|_{/\underline{\leftrightarrow}}$ is finite.

Theorem (Normal form of countable languages)

Given a μ–automaton A, if $\mathcal{L}(A)$ is countable up to bisimulation, then it can be represented by

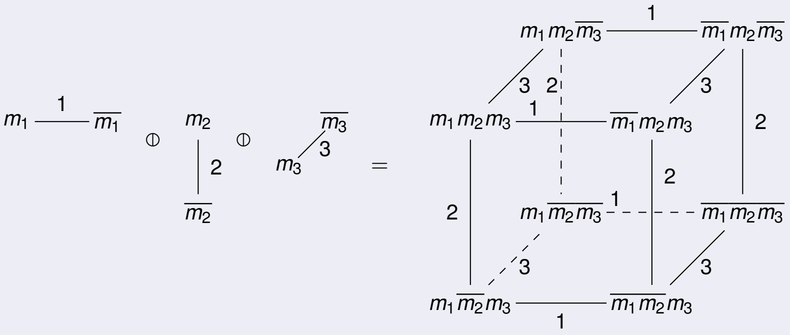
$$F_n[x_1\backslash\mathcal{T}_1, \ldots, x_1\backslash\mathcal{T}_n]$$

for some $n < \omega$, $\{\mathcal{T}_1, \ldots, \mathcal{T}_n\} \subseteq alive(\mathcal{L}(A))$, and some $F_n \subseteq \mathfrak{T}_n$ which is recognizable by an finite automaton B on finite trees in $\mathfrak{T}_n$.

Model Abstraction

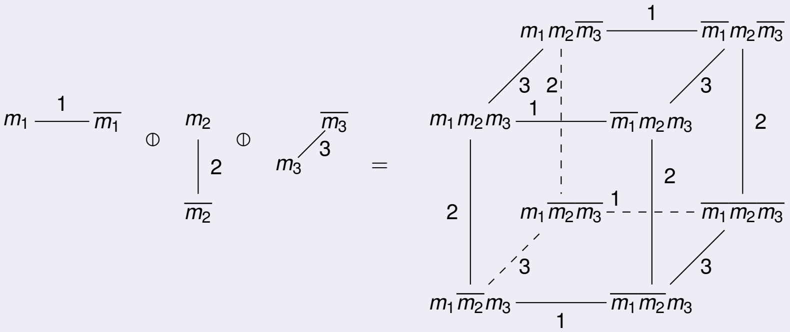
Model Abstraction

State space explosion: e.g., n children, 2^n states.



Model Abstraction

State space explosion: e.g., n children, 2^n states.



Good representations may help.

More than 10^{20} states can be handled by *symbolic model checking* [BCM$^+$92].

Model Abstraction

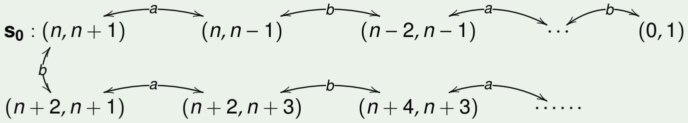
However, it is never enough...

Model Abstraction

However, it is never enough...

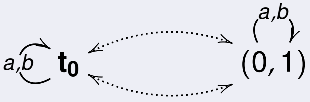
Example (Guessing the other number)

a and b are given two natural numbers n and $n+1$ respectively. They are told that what they have are two consecutive natural numbers, but they do not know who has the bigger one. We can build the following model (suppose n is an even number):



Model Abstraction

Making the models smaller by abstraction



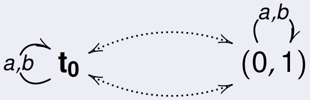
Clearly, we may lose some information...

Model abstraction

Idea: safely reason about the big models at their small abstractions.

Model Abstraction

Making the models smaller by abstraction



Clearly, we may lose some information...

Model abstraction

Idea: safely reason about the big models at their small abstractions.
"Most general [technique to reduce state space] and flexible" [CGL94]. Can be fully automated [CGJ+03].

Abstraction w.r.t. PAL: with Dechesne & Orzan

Definition (Kripke Modal Labelled Transition System [HJS01])

A *Kripke Modal Labelled Transition System* (KMLTS) is a tuple $\mathcal{M} = (S, \mathbf{P}, \mathbf{\Sigma}, \dashrightarrow, \rightarrow, V)$ where:

- $\dashrightarrow$ is a set of transitions of the form $s \overset{i}{\dashrightarrow} s'$ where $i \in \mathbf{\Sigma}$;
- $\rightarrow$ is a set of transitions of the form $s \overset{i}{\rightarrow} s'$ where $i \in \mathbf{\Sigma}$;
- V is a valuation function: $V : S \rightarrow \{true, false, \uparrow\}^{\mathbf{P}}$.

We require that $\rightarrow \subseteq \dashrightarrow$.

Abstraction w.r.t. PAL: with Dechesne & Orzan

Definition (Kripke Modal Labelled Transition System [HJS01])

A *Kripke Modal Labelled Transition System* (KMLTS) is a tuple $\mathcal{M} = (S, \mathbf{P}, \mathbf{\Sigma}, \dashrightarrow, \rightarrow, V)$ where:

- $\dashrightarrow$ is a set of transitions of the form $s \overset{i}{\dashrightarrow} s'$ where $i \in \mathbf{\Sigma}$;
- $\rightarrow$ is a set of transitions of the form $s \overset{i}{\rightarrow} s'$ where $i \in \mathbf{\Sigma}$;
- V is a valuation function: $V : S \rightarrow \{true, false, \uparrow\}^{\mathbf{P}}$.

We require that $\rightarrow \subseteq \dashrightarrow$.

The signature of $\mathcal{M}$: $(\mathbf{P}, \mathbf{\Sigma})$ is also important.

Model Abstraction

Recall the formulas of the Public Announcement Logic:

$$\phi \quad ::= \quad p \mid \phi \wedge \psi \mid \neg\phi \mid \Box_i\phi \mid [!\phi]\phi$$

Model Abstraction

Recall the formulas of the Public Announcement Logic:

$$\phi \quad ::= \quad p \mid \phi \wedge \psi \mid \neg\phi \mid \Box_i\phi \mid [!\phi]\phi$$

3-valued semantics:

$$[\![\Box_i\phi]\!]^{\mathcal{M},s} \quad = \quad \begin{cases} \mathit{true} & \text{if } \forall s' : s \overset{i}{\dashrightarrow} s' \implies [\![\phi]\!]^{\mathcal{M},s'} = \mathit{true} \\ \mathit{false} & \text{if } \exists s' : s \overset{i}{\rightarrow} s' \text{ and } [\![\phi]\!]^{\mathcal{M},s'} = \mathit{false} \\ \uparrow & \text{otherwise} \end{cases}$$

Model Abstraction

Recall the formulas of the Public Announcement Logic:

$$\phi \quad ::= \quad p \mid \phi \wedge \psi \mid \neg\phi \mid \Box_i\phi \mid [!\phi]\phi$$

3-valued semantics:

$$[\![\Box_i\phi]\!]^{\mathcal{M},s} = \begin{cases} \mathit{true} & \text{if } \forall s' : s \overset{i}{\dashrightarrow} s' \implies [\![\phi]\!]^{\mathcal{M},s'} = \mathit{true} \\ \mathit{false} & \text{if } \exists s' : s \overset{i}{\longrightarrow} s' \text{ and } [\![\phi]\!]^{\mathcal{M},s'} = \mathit{false} \\ \uparrow & \text{otherwise} \end{cases}$$

$$[\![[!\phi]\psi]\!]^{\mathcal{M},s} = \begin{cases} \mathit{true} & \text{if } [\![\phi]\!]^{\mathcal{M},s} = \mathit{false} \text{ or } [\![\psi]\!]^{\mathcal{M}|_\phi,s} = \mathit{true} \\ \mathit{false} & \text{if } [\![\phi]\!]^{\mathcal{M},s} = \mathit{true} \text{ and } [\![\psi]\!]^{\mathcal{M}|_\phi,s} = \mathit{false} \\ \uparrow & \text{otherwise} \end{cases}$$

Refinement and Abstraction

A relation linking the abstract to the more refined

$$\mathcal{N}, t : \mathbf{I}', P' \qquad \Subset_{f,g} \qquad \mathcal{M}, s : \mathbf{I}, P$$

Refinement and Abstraction

A relation linking the abstract to the more refined

$$\mathcal{N}, t : \mathbf{I}', P' \qquad \Subset_{f,g} \qquad \mathcal{M}, s : \mathbf{I}, P$$

$$1, 2 \qquad \mapsto_f \qquad c$$

Refinement and Abstraction

A relation linking the abstract to the more refined

$\mathcal{N}, t : \mathbf{I}', P'$	$\Subset_{f,g}$	$\mathcal{M}, s : \mathbf{I}, P$
$1, 2$	$\mapsto_f$	c
p_1, p_2	$\mapsto_g$	p_c

Refinement and Abstraction

A relation linking the abstract to the more refined

$\mathcal{N}, t : \mathbf{I}', P'$	$\Subset_{f,g}$	$\mathcal{M}, s : \mathbf{I}, P$
$1, 2$	$\mapsto_f$	c
p_1, p_2	$\mapsto_g$	p_c
$\bullet$		$\bullet$

Refinement and Abstraction

A relation linking the abstract to the more refined

$\mathcal{N}, t : \mathbf{I}', P'$	$\Subset_{f,g}$	$\mathcal{M}, s : \mathbf{I}, P$
$1, 2$	$\mapsto_f$	c
p_1, p_2	$\mapsto_g$	p_c
$\bullet$		$\bullet\ p_c : \mathit{true}$

Refinement and Abstraction

A relation linking the abstract to the more refined

$\mathcal{N}, t : \mathbf{I}', P'$	$\Subset_{f,g}$	$\mathcal{M}, s : \mathbf{I}, P$
$1, 2$	$\mapsto_f$	c
p_1, p_2	$\mapsto_g$	p_c
$\bullet\ p_1, p_2 : true$		$\bullet\ p_c : true$

Refinement and Abstraction

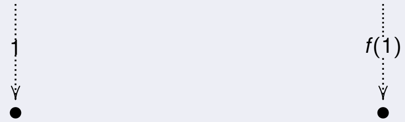
A relation linking the abstract to the more refined

$\mathcal{N}, t : \mathbf{I}', P'$	$\Subset_{f,g}$	$\mathcal{M}, s : \mathbf{I}, P$
$1, 2$	$\mapsto_f$	c
p_1, p_2	$\mapsto_g$	p_c
$\bullet\ p_1, p_2 : true$		$\bullet\ p_c : true$

Refinement and Abstraction

A relation linking the abstract to the more refined

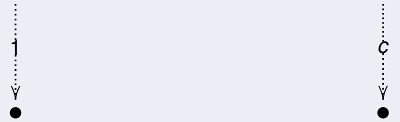
$\mathcal{N}, t : \mathbf{I}', P'$	$\Subset_{f,g}$	$\mathcal{M}, s : \mathbf{I}, P$
$1, 2$	$\mapsto_f$	c
p_1, p_2	$\mapsto_g$	p_c
$\bullet\ p_1, p_2 : true$		$\bullet\ p_c : true$



Refinement and Abstraction

A relation linking the abstract to the more refined

$\mathcal{N}, t : \mathbf{I}', P'$	$\Subset_{f,g}$	$\mathcal{M}, s : \mathbf{I}, P$
$1, 2$	$\mapsto_f$	c
p_1, p_2	$\mapsto_g$	p_c
$\bullet\ p_1, p_2 : true$		$\bullet\ p_c : true$



Refinement and Abstraction

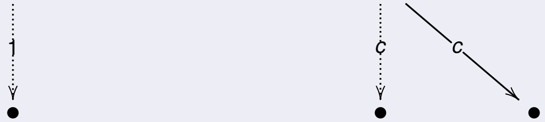
A relation linking the abstract to the more refined

$\mathcal{N}, t : \mathbf{I}', P'$	$\Subset_{f,g}$	$\mathcal{M}, s : \mathbf{I}, P$
$1, 2$	$\mapsto_f$	c
p_1, p_2	$\mapsto_g$	p_c
$\bullet\ p_1, p_2 : true$		$\bullet\ p_c : true$

Refinement and Abstraction

A relation linking the abstract to the more refined

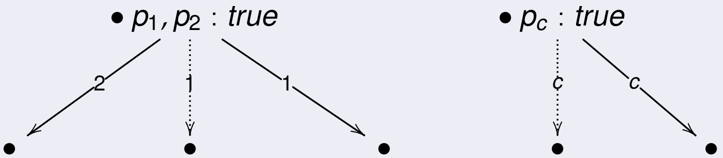
$\mathcal{N}, t : \mathbf{I}', P'$	$\Subset_{f,g}$	$\mathcal{M}, s : \mathbf{I}, P$
$1, 2$	$\mapsto_f$	c
p_1, p_2	$\mapsto_g$	p_c
$\bullet\ p_1, p_2 : true$		$\bullet\ p_c : true$



Refinement and Abstraction

A relation linking the abstract to the more refined

$\mathcal{N}, t : \mathbf{I}', P'$	$\Subset_{f,g}$	$\mathcal{M}, s : \mathbf{I}, P$
$1, 2$	$\mapsto_f$	c
p_1, p_2	$\mapsto_g$	p_c



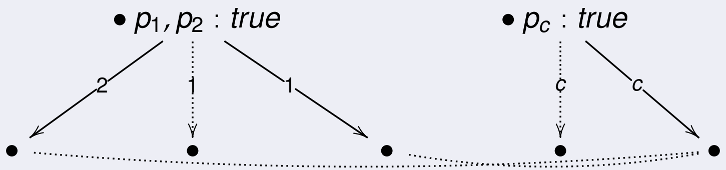
Refinement and Abstraction

A relation linking the abstract to the more refined

$$\mathcal{N}, t : \mathbf{I}', P' \quad \Subset_{f,g} \quad \mathcal{M}, s : \mathbf{I}, P$$

$$1, 2 \quad \mapsto_f \quad c$$

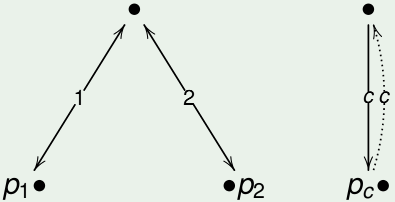
$$p_1, p_2 \quad \mapsto_g \quad p_c$$



Model Abstraction

Example

Example of a KMLTS $\mathcal{M}$ and a f, g–abstraction of it where $f(1) = f(2) = c; g(p_1) = g(p_2) = p_c$.



Desired Property

Notation

Given two pointed models $(\mathcal{M}, s), (\mathcal{N}, t)$, and two formulas ϕ, ψ, we say $[\![\psi]\!]^{\mathcal{M},s} \leq [\![\phi]\!]^{\mathcal{N},t}$ if the following hold:

1. $[\![\psi]\!]^{\mathcal{M},s} = \mathit{true} \implies [\![\phi]\!]^{\mathcal{N},t} = \mathit{true}$;
2. $[\![\psi]\!]^{\mathcal{M},s} = \mathit{false} \implies [\![\phi]\!]^{\mathcal{N},t} = \mathit{false}$.

Desired Property

Notation

Given two pointed models $(\mathcal{M}, s), (\mathcal{N}, t)$, and two formulas ϕ, ψ, we say $[\![\psi]\!]^{\mathcal{M},s} \leq [\![\phi]\!]^{\mathcal{N},t}$ if the following hold:

1. $[\![\psi]\!]^{\mathcal{M},s} = \mathit{true} \implies [\![\phi]\!]^{\mathcal{N},t} = \mathit{true}$;
2. $[\![\psi]\!]^{\mathcal{M},s} = \mathit{false} \implies [\![\phi]\!]^{\mathcal{N},t} = \mathit{false}$.

Safe reasoning

$(\mathcal{N}, t) \Subset_{f,g} (\mathcal{M}, s)$ implies for all $\phi \in \mathtt{PAL}_{\mathbf{I}',P'}$:
$[\![\ulcorner\phi\urcorner_{f,g}]\!]^{\mathcal{M},s} \leq [\![\phi]\!]^{\mathcal{N},t}$.

Translation of the formulas

Definition (Translation of formulas)

Given signatures $(\mathbf{I}', P'), (\mathbf{I}, P)$, and surjective functions $f : \mathbf{I}' \to \mathbf{I}, g : P' \to P$, we define the translation of an $\mathtt{PAL}_{\mathbf{I}',P'}$-formula ϕ into an $\mathtt{PAL}_{\mathbf{I},P}$-formula $\ulcorner \phi \urcorner_{f,g}$ inductively as follows:

$$\begin{array}{lcl}
\ulcorner p' \urcorner_{f,g} & = & g(p') \\
\ulcorner \neg\psi \urcorner_{f,g} & = & \neg \ulcorner \psi \urcorner_{f,g} \\
\ulcorner \psi_1 \wedge \psi_2 \urcorner_{f,g} & = & \ulcorner \psi_1 \urcorner_{f,g} \wedge \ulcorner \psi_2 \urcorner_{f,g} \\
\ulcorner K_{i'} \psi \urcorner_{f,g} & = & K_{f(i')} \ulcorner \psi \urcorner_{f,g} \\
\ulcorner [\chi]\psi \urcorner_{f,g} & = & [\ulcorner \chi \urcorner_{f,g}] \ulcorner \psi \urcorner_{f,g}
\end{array}$$

Translation of the formulas

Definition (Translation of formulas)

Given signatures $(\mathbf{I}', P'), (\mathbf{I}, P)$, and surjective functions $f : \mathbf{I}' \to \mathbf{I}, g : P' \to P$, we define the translation of an $\mathtt{PAL}_{\mathbf{I}',P'}$-formula ϕ into an $\mathtt{PAL}_{\mathbf{I},P}$-formula $\ulcorner\phi\urcorner_{f,g}$ inductively as follows:

$$
\begin{array}{lcl}
\ulcorner p' \urcorner_{f,g} & = & g(p') \\
\ulcorner \neg\psi \urcorner_{f,g} & = & \neg\ulcorner \psi \urcorner_{f,g} \\
\ulcorner \psi_1 \wedge \psi_2 \urcorner_{f,g} & = & \ulcorner \psi_1 \urcorner_{f,g} \wedge \ulcorner \psi_2 \urcorner_{f,g} \\
\ulcorner K_{i'}\psi \urcorner_{f,g} & = & K_{f(i')}\ulcorner \psi \urcorner_{f,g} \\
\ulcorner [\chi]\psi \urcorner_{f,g} & = & [\ulcorner \chi \urcorner_{f,g}]\ulcorner \psi \urcorner_{f,g}
\end{array}
$$

Example

$\ulcorner [p \wedge q \wedge r]K_1 p \vee K_2 q \urcorner_{f,g} = [P \wedge R]K_A P$ with $f(1) = f(2) = A$; $g(p) = g(q) = P$ and $g(r) = R$.

Logical Characterization

Lemma

$(\mathcal{N}, t) \Subset_{f,g} (\mathcal{M}, s)$ implies $(\mathcal{N}|_{\chi}, t) \Subset_{f,g} (\mathcal{M}|_{\ulcorner \chi \urcorner_{f,g}}, s)$ under certain condition.

Logical Characterization

Lemma

$(\mathcal{N}, t) \Subset_{f,g} (\mathcal{M}, s)$ *implies* $(\mathcal{N}|_{\chi}, t) \Subset_{f,g} (\mathcal{M}|_{\ulcorner \chi \urcorner_{f,g}}, s)$ *under certain condition.*

Theorem

$(\mathcal{N}, t) \Subset_{f,g} (\mathcal{M}, s)$ *implies for all* $\phi \in \mathtt{PAL}_{I', P'}$:
$[\![\ulcorner \phi \urcorner_{f,g}]\!]^{\mathcal{M},s} \leq [\![\phi]\!]^{\mathcal{N},t}$.

Logical Characterization

Lemma

$(\mathcal{N}, t) \Subset_{f,g} (\mathcal{M}, s)$ implies $(\mathcal{N}|_{\chi}, t) \Subset_{f,g} (\mathcal{M}|_{\ulcorner\chi\urcorner_{f,g}}, s)$ under certain condition.

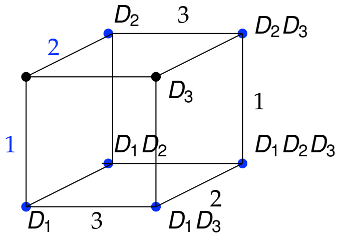
Theorem

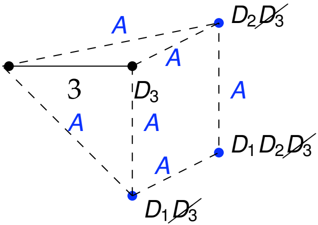
$(\mathcal{N}, t) \Subset_{f,g} (\mathcal{M}, s)$ implies for all $\phi \in PAL_{I',P'}$: $[\![\ulcorner\phi\urcorner_{f,g}]\!]^{\mathcal{M},s} \leq [\![\phi]\!]^{\mathcal{N},t}$.

Theorem

If for every formula $\phi \in PAL_{I',P'}$: $[\![\ulcorner\phi\urcorner_{f,g}]\!]^{\mathcal{M},s} \leq [\![\phi]\!]^{\mathcal{N},t}$ then $(\mathcal{N}, t) \Subset_{f,g} (\mathcal{M}, s)$ (Image finitness assumed).

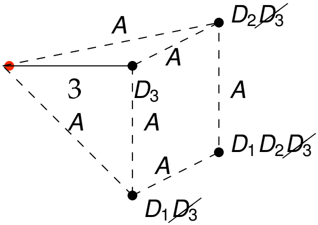
Muddy Children - Abstraction of n=3 case





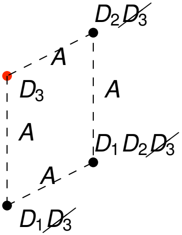
Abstractions of the Muddy Children for $n = 3$ children. $f(1) = f(2) = A, f(3) = 3$ and $g = Id$. $\not{D}_3$ means proposition D_3 has valuation $\perp$ in the current state.

Muddy Children - Abstraction of n=3 case



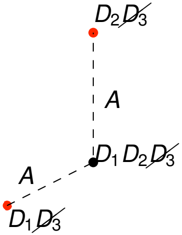
First announcement: $\ulcorner D_1 \vee D_2 \vee D_3 \urcorner_{f,g} = D_1 \vee D_2 \vee D_3$.

Muddy Children - Abstraction of n=3 case



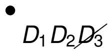
$\ulcorner K_3 D_3 \urcorner_{f,g} = K_3 D_3$ holds at world D_3. Announcement can be made if more than one child is dirty:
$\ulcorner \neg K_1 D_1 \wedge \neg K_2 D_2 \wedge \neg K_3 D_3 \urcorner_{f,g} = \neg K_A D_1 \wedge \neg K_A D_2 \wedge \neg K_3 D_3$.

Muddy Children - Abstraction of n=3 case



$\ulcorner K_1 D_1 \urcorner_{f,g} = K_A D_1$ holds at world $D_1 \not{D}_3$ and $\ulcorner K_1 D_2 \urcorner_{f,g} = K_A D_2$ holds at $D_2 \not{D}_3$. If all the three children are dirty then announce: $\ulcorner \neg K_1 D_1 \wedge \neg K_2 D_2 \wedge \neg K_3 D_3 \urcorner_{f,g} = \neg K_A D_1 \wedge \neg K_A D_2 \wedge \neg K_3 D_3$.

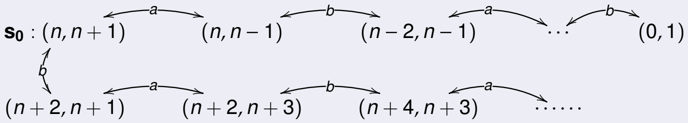
Muddy Children - Abstraction of n=3 case



$\ulcorner K_1 D_1 \wedge K_2 D_2 \urcorner_{f,g} = K_A D_1 \wedge K_A D_2$ holds at the only world.

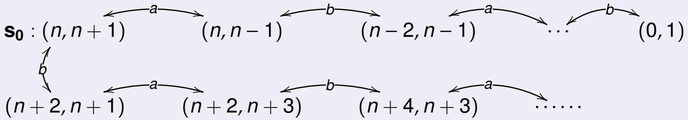
Model Abstraction

Recall the *Guessing Number* example

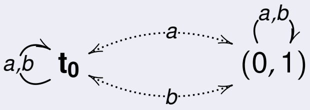


Model Abstraction

Recall the *Guessing Number* example



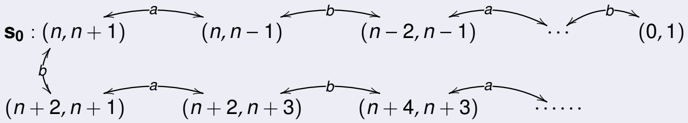
Can we do better than this?



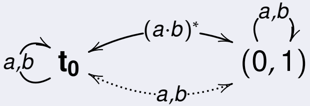
Not enough *must* transitions for evaluating existential formulas (properties of reachability) e.g., $\langle(a+b)^*\rangle has_a 0$.

Model Abstraction

Recall the *Guessing Number* example

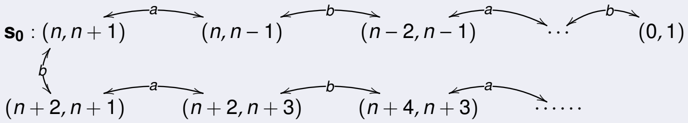


Yes! *Accelerated Modal LTS* [EvdP06]

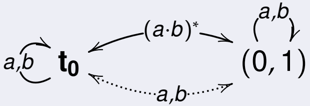


Model Abstraction

Recall the *Guessing Number* example



Yes! *Accelerated Modal LTS* [EvdP06]



$\langle\pi\rangle\phi$ is true at s if there is a *must*-path $s \xrightarrow{\pi_1} s_1 \xrightarrow{\pi_2} \cdots \xrightarrow{\pi_n} s_n$ to a state where ϕ is true, such that $\mathcal{L}(\pi_1 \cdots \pi_n) \subseteq \mathcal{L}(\pi)$. e.g, $\langle(a+b)^*\rangle has_a 0$ is true on the above abstract model.

PDL on Accelerated Models: with Chen & van de Pol

Question: how to model check PDL on AMLTS?

PDL on Accelerated Models: with Chen & van de Pol

Question: how to model check PDL on AMLTS?

Recall the semantics:

$$\mathcal{M}, s \models \langle \pi \rangle \phi \quad \Leftrightarrow \quad \text{there exists a path } s \xrightarrow{\pi_1} s_1 \xrightarrow{\pi_2} \cdots \xrightarrow{\pi_n} s_n : \mathcal{M}, s_n \models \phi \text{ and } \mathcal{L}(\pi_1 \cdots \pi_n) \subseteq \mathcal{L}(\pi)$$

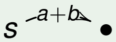
PDL on Accelerated Models: with Chen & van de Pol

Question: how to model check PDL on AMLTS?

Recall the semantics:

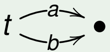
$$\mathcal{M}, s \vDash \langle\pi\rangle\phi \quad \Leftrightarrow \quad \text{there exists a path } s \xrightarrow{\pi_1} s_1 \xrightarrow{\pi_2} \cdots \xrightarrow{\pi_n} s_n : \; \mathcal{M}, s_n \vDash \phi \text{ and } \mathcal{L}(\pi_1 \cdots \pi_n) \subseteq \mathcal{L}(\pi)$$

Example



$\mathcal{M}, s \vDash \langle a+b+c \rangle \top$

$\mathcal{M}, s \nvDash \langle a \rangle \top$



$\mathcal{M}, t \vDash \langle a+b+c \rangle \top$

$\mathcal{M}, t \vDash \langle a \rangle \top \wedge \langle b \rangle \top$

A straightforward idea which is hard to implement: reduce the new to the old.

PDL on Accelerated Models

Regular Expression Rewriting [CDGLV02]

Given a regular expression π, rewrite π, if possible, by a set of other regular expressions $\mathcal{E} = \{\pi_0, \pi_1, \ldots \pi_n\}$.

PDL on Accelerated Models

Regular Expression Rewriting [CDGLV02]

Given a regular expression π, rewrite π, if possible, by a set of other regular expressions $\mathcal{E} = \{\pi_0, \pi_1, \ldots \pi_n\}$. For example, we can rewrite $a^* \cdot d$ by $\{a \cdot a^*, d\}$: $a \cdot a^* \cdot d + d$.

PDL on Accelerated Models

Regular Expression Rewriting [CDGLV02]

Given a regular expression π, rewrite π, if possible, by a set of other regular expressions $\mathcal{E} = \{\pi_0, \pi_1, \ldots \pi_n\}$. For example, we can rewrite $a^* \cdot d$ by $\{a \cdot a^*, d\}$: $a \cdot a^* \cdot d + d$. In most of the cases, we do not have an exact rewriting. e.g., try to rewrite $a + b$ by $\{a\}$. In such cases we are interested in the *maximal* rewriting.

PDL on Accelerated Models

Regular Expression Rewriting [CDGLV02]

Given a regular expression π, rewrite π, if possible, by a set of other regular expressions $\mathcal{E} = \{\pi_0, \pi_1, \ldots \pi_n\}$. For example, we can rewrite $a^* \cdot d$ by $\{a \cdot a^*, d\}$: $a \cdot a^* \cdot d + d$. In most of the cases, we do not have an exact rewriting. e.g., try to rewrite $a + b$ by $\{a\}$. In such cases we are interested in the *maximal* rewriting.

PDL on Accelerated Models

Regular Expression Rewriting [CDGLV02]

Given a regular expression π, rewrite π, if possible, by a set of other regular expressions $\mathcal{E} = \{\pi_0, \pi_1, \ldots \pi_n\}$. For example, we can rewrite $a^* \cdot d$ by $\{a \cdot a^*, d\}$: $a \cdot a^* \cdot d + d$. In most of the cases, we do not have an exact rewriting. e.g., try to rewrite $a + b$ by $\{a\}$. In such cases we are interested in the *maximal* rewriting.

Theorem

([CDGLV02]) There is an essentially optimal algorithm to compute the maximal $\mathcal{E}$-rewriting of a given π w.r.t a given set $\mathcal{E}$ in 2-EXPTIME*.*

$\mathtt{PDL}$ on Accelerated Models

Definition (Rewriting $\mathtt{PDL}$ formula w.r.t. an accelerated model)

Given an AKM $\mathcal{M}$ and a $\mathtt{PDL}_{\mathbf{\Sigma}}$ formula ϕ, let $\langle\rangle_{\mathcal{M}}$ be the set of labels in $\mathcal{M}$, $\Re_{\mathcal{M}}(\phi)$ is the rewriting of ϕ in the language $\mathtt{PDL}_{\mathbf{\Sigma}_{\langle\rangle_{\mathcal{M}}}}$ defined by:

- $\Re_{\mathcal{M}}(\langle\pi\rangle\psi) = \langle\widehat{\pi_{\langle\rangle_{\mathcal{M}}}}\rangle\Re_{\mathcal{M}}(\psi)$.

PDL on Accelerated Models

Definition (Rewriting PDL formula w.r.t. an accelerated model)

Given an AKM $\mathcal{M}$ and a $\mathtt{PDL}_{\mathbf{\Sigma}}$ formula ϕ, let $\langle\rangle_{\mathcal{M}}$ be the set of labels in $\mathcal{M}$, $\Re_{\mathcal{M}}(\phi)$ is the rewriting of ϕ in the language $\mathtt{PDL}_{\mathbf{\Sigma}_{\langle\rangle_{\mathcal{M}}}}$ defined by:

- $\Re_{\mathcal{M}}(\langle\pi\rangle\psi) = \langle\widehat{\pi_{\langle\rangle_{\mathcal{M}}}}\rangle\Re_{\mathcal{M}}(\psi)$.

Theorem

For any pointed AKM $\mathcal{M}, s$ and any $\mathtt{PDL}_{\mathbf{\Sigma}}$ formula ϕ,

$$\mathcal{M}, s \vDash \phi \iff \ulcorner\mathcal{M}\urcorner, s \Vdash \Re_{\mathcal{M}}(\phi).$$

PDL on Accelerated Models

We also give a direct model checking algorithm by using the following proposition:

Theorem

Given a pointed AKM $\mathcal{M} = (S, \mathbf{\Sigma}, \rightarrow, V, s_0)$ *and* $T \subseteq S$*, if* $T = \{t \mid \mathcal{M}, t \vDash \phi\}$*, then we have:*

$$\mathcal{M}, s_0 \models \langle\pi\rangle\phi \iff \mathcal{L}(\mathcal{M} \otimes_T A_\pi) \neq \emptyset$$

where A_π *denotes the deterministic automaton corresponding to* π*.*

PDL on Accelerated Models

Theorem

We can reduce the problem of the existence of the non-empty rewriting to model checking problem.

$$\textit{there is a non-empty rewriting of } \pi \textit{ w.r.t. } \mathcal{E} \iff$$

PDL on Accelerated Models

Theorem

We can reduce the problem of the existence of the non-empty rewriting to model checking problem.

$$\textit{there is a non-empty rewriting of } \pi \textit{ w.r.t. } \mathcal{E} \iff \mathcal{M}_{\mathcal{E}}, s \vDash \langle \pi \rangle \phi.$$

PDL on Accelerated Models

Theorem

We can reduce the problem of the existence of the non-empty rewriting to model checking problem.

$$\textit{there is a non-empty rewriting of } \pi \textit{ w.r.t. } \mathcal{E} \iff \mathcal{M}_{\mathcal{E}}, s \models \langle\pi\rangle\phi.$$

Theorem

Model checking PDL on AKM is EXPSPACE*-complete.*

$\mathtt{PDL}$ on Accelerated Models

Definition

Rewriting of a $\mathtt{PDL}^+_{\mathbf{\Sigma}}$ formula Given a $\mathtt{PDL}^+_{\mathbf{\Sigma}}$ formula ϕ, let $\langle\rangle_\phi$ be the set $\{\pi \mid \langle\pi\rangle \text{ appears in } \phi\}$, $\Re(\phi)$ is the rewriting of ϕ in the language $\mathtt{PDL}^+_{\mathbf{\Sigma}_{\langle\rangle_\phi}}$ defined by:

- $\Re(\langle\pi\rangle(\psi)) = \langle e_\pi\rangle\Re(\psi)$.
- $\Re([\pi]\psi) = [\widehat{\pi_{\langle\rangle_\phi}}]\Re(\psi)$.

PDL on Accelerated Models

Definition

Rewriting of a $\mathtt{PDL}^{+}_{\mathbf{\Sigma}}$ formula Given a $\mathtt{PDL}^{+}_{\mathbf{\Sigma}}$ formula ϕ, let $\langle\rangle_\phi$ be the set $\{\pi \mid \langle\pi\rangle \text{ appears in } \phi\}$, $\Re(\phi)$ is the rewriting of ϕ in the language $\mathtt{PDL}^{+}_{\mathbf{\Sigma}_{\langle\rangle_\phi}}$ defined by:

- $\Re(\langle\pi\rangle(\psi)) = \langle e_\pi\rangle\Re(\psi)$.
- $\Re([\pi]\psi) = [\widehat{\pi_{\langle\rangle_\phi}}]\Re(\psi)$.

Theorem

Given a $\mathtt{PDL}^{+}_{\mathbf{\Sigma}}$ formula ϕ, ϕ is satisfiable on an AKM $\iff$ $\Re(\phi)$ is satisfiable on a standard Kripke model.

Please contact me if you are interested in solving them...

Please contact me if you are interested in solving them...

- The succinctness of the newly introduced protocol logics

Please contact me if you are interested in solving them...

- The succinctness of the newly introduced protocol logics
- A version of 3-valued full DEL

Please contact me if you are interested in solving them...

- The succinctness of the newly introduced protocol logics
- A version of 3-valued full DEL
- Model composition with agent expansion

Please contact me if you are interested in solving them...

- The succinctness of the newly introduced protocol logics
- A version of 3-valued full DEL
- Model composition with agent expansion
- Rewriting a modal formula by other formulas

Please contact me if you are interested in solving them...

- The succinctness of the newly introduced protocol logics
- A version of 3-valued full DEL
- Model composition with agent expansion
- Rewriting a modal formula by other formulas
- Counting S5 model modulo bisimulation

Please contact me if you are interested in solving them...

- The succinctness of the newly introduced protocol logics
- A version of 3-valued full DEL
- Model composition with agent expansion
- Rewriting a modal formula by other formulas
- Counting S5 model modulo bisimulation
- An epistemic logic of Cryptography

Please contact me if you are interested in solving them...

- The succinctness of the newly introduced protocol logics
- A version of 3-valued full DEL
- Model composition with agent expansion
- Rewriting a modal formula by other formulas
- Counting S5 model modulo bisimulation
- An epistemic logic of Cryptography
- Is the derived protocol (as a set of possible sequences) at a (finite) Kripke model over a finite set of action models always regular?

Please contact me if you are interested in solving them...

- The succinctness of the newly introduced protocol logics
- A version of 3-valued full $\mathtt{DEL}$
- Model composition with agent expansion
- Rewriting a modal formula by other formulas
- Counting S5 model modulo bisimulation
- An epistemic logic of Cryptography
- Is the derived protocol (as a set of possible sequences) at a (finite) Kripke model over a finite set of action models always regular?
- Let $\mathtt{DEL}^*$ be the language of $\mathtt{DEL}$with the new modality $< A^* >$. Can you fit this extended logic into a suitable fixed point logic?

Please contact me if you are interested in solving them...

- The succinctness of the newly introduced protocol logics
- A version of 3-valued full DEL
- Model composition with agent expansion
- Rewriting a modal formula by other formulas
- Counting S5 model modulo bisimulation
- An epistemic logic of Cryptography
- Is the derived protocol (as a set of possible sequences) at a (finite) Kripke model over a finite set of action models always regular?
- Let $\mathtt{DEL}^*$ be the language of DELwith the new modality $< A^* >$. Can you fit this extended logic into a suitable fixed point logic?
- Is the model checking problem of $\mathtt{DEL}^*$ decidable?

Please contact me if you are interested in solving them...

- The succinctness of the newly introduced protocol logics
- A version of 3-valued full DEL
- Model composition with agent expansion
- Rewriting a modal formula by other formulas
- Counting S5 model modulo bisimulation
- An epistemic logic of Cryptography
- Is the derived protocol (as a set of possible sequences) at a (finite) Kripke model over a finite set of action models always regular?
- Let $\mathtt{DEL}^*$ be the language of DELwith the new modality $< A^* >$. Can you fit this extended logic into a suitable fixed point logic?
- Is the model checking problem of $\mathtt{DEL}^*$ decidable?

Thank you very much for your attention!

J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, and L. J. Hwang.
Symbolic model checking: 10^{20} states and beyond.
Information and Computation, 98(2):142–170, June 1992.

J. A. Brzozowski.
Derivatives of regular expressions.
Journal of the ACM, 11(4):481–494, October 1964.

D. Calvanese, G. De Giacomoa, M. Lenzerinia, and M. Y. Vardi.
Rewriting of regular expressions and regular path queries.
Journal of Computer and System Sciences, 64(3):443–465, May 2002.

E. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith.
Counterexample-guided abstraction refinement for symbolic model checking.
Journal of the ACM, 50(5):752–794, September 2003.

E. M. Clarke, O. Grumberg, and D. E. Long.
Model checking and abstraction.
ACM Transactions on Programming Languages and Systems, 16(5):1512–1542, September 1994.

J. H. Conway.
Regular Algebra and Finite Machines.
Chapman and Hall, September 1971.

D. Dams and K. S. Namjoshi.
Automata as abstractions.
In *Proceedings of VMCAI '05*, pages 216–232, 2005.

M. Espada and J. van de Pol.
Accelerated modal abstractions of labelled transition systems.
In M. J.son and Varmo Vene, editors, *Proceedings of AMAST '06*, volume 4019, pages 338–352, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

M. J. Fischer and R. E. Ladner.
Propositional dynamic logic of regular programs.
Journal of Computer and System Sciences, 18(2):194–211, 1979.

J. Gerbrandy and W. Groeneveld.
Reasoning about information change.
Journal of Logic, Language and Information, 6(2):147–169, April 1997.

S. Hart.
“knowing whether”, “knowing that”, and the cardinality of state spaces.
Journal of Economic Theory, 70(1):249–256, July 1996.

J. Y. Halpern and R. Fagin.
Modelling knowledge and action in distributed systems.
Distributed Computing, 3(4):159–177, 1989.

M. Huth, R. Jagadeesan, and D. Schmidt.

Modal transition systems: A foundation for three-valued program analysis.
In D. Sands, editor, *Programming Languages and Systems*, volume 2028, pages 155–169, Berlin, Heidelberg, March 2001. Springer Berlin Heidelberg.

T. Hoshi.
Epistemic Dynamics and Protocol Information.
PhD thesis, Stanford, 2009.

T. Hoshi and A. Yap.
Dynamic epistemic logic with branching temporal structures.
Synthese, 169(2):259–281, July 2009.

D. Janin and I. Walukiewicz.
Automata for the modal μ-calculus and related results.
In *Proceedings of MFCS '95*, pages 552–562, 1995.

D. Kozen.

Results on the propositional μ-calculus.
Theoretical Computer Science, 27(3):333–354, 1983.

D. Kozen.
Automata on guarded strings and applications.
Technical report, Ithaca, NY, USA, 2001.

D. Niwiński.
On the cardinality of sets of infinite trees recognizable by finite automata.
In *Proceedings of MFCS '91*, pages 367–376, 1991.

J. A. Plaza.
Logics of public communications.
In M. L. Emrich, M. S. Pfeifer, M. Hadzikadic, and Z. W. Ras, editors, *Proceedings of the 4th International Symposium on Methodologies for Intelligent Systems*, pages 201–216, 1989.

R. Parikh and R. Ramanujam.

A knowledge based semantics of messages.
Journal of Logic, Language and Information, 12(4), 2003.

J. van Benthem, J. Gerbrandy, T. Hoshi, and E Pacuit.
Merging frameworks for interaction.
Journal of Philosophical Logic, 38(5):491–526, October 2009.

J. van Benthem and D. Ikegami.
Modal fixed-point logic and changing models.
In *Pillars of Computer Science*, pages 146–165, 2008.