Term Modal Logic

Anantha Padmanabha

joint work with R. Ramanujam

Institute of Mathematical Sciences, HBNI, Chennai

TU, Munich
3 September 2019

Propositional Multi Modal logic

- Propositional Multi-modal logic is extensively used in various fields of theoretical computer science.

Propositional Multi Modal logic

- Propositional Multi-modal logic is extensively used in various fields of theoretical computer science.
- Syntax: Fix a finite index set $Ag = \{1 \cdots n\}$. Let $\mathcal{P}$ be a countable set of propositions.

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Box_i \varphi$$

where $p \in \mathcal{P}$ and $i \in Ag$.

Propositional Multi Modal logic

- Propositional Multi-modal logic is extensively used in various fields of theoretical computer science.
- Syntax: Fix a finite index set $Ag = \{1 \cdots n\}$. Let $\mathcal{P}$ be a countable set of propositions.

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Box_i\, \varphi$$

  where $p \in \mathcal{P}$ and $i \in Ag$.
- Kripke structure: $\mathcal{M} = (\mathcal{W}, \mathcal{R}_1, \ldots R_n, \rho)$ is a structure where: $R_i \subseteq (\mathcal{W} \times \mathcal{W})$ and $\rho : \mathcal{W} \to 2^{\mathcal{P}}$.

Propositional Multi Modal logic

- Propositional Multi-modal logic is extensively used in various fields of theoretical computer science.
- Syntax: Fix a finite index set $Ag = \{1 \cdots n\}$. Let $\mathcal{P}$ be a countable set of propositions.

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Box_i\, \varphi$$

  where $p \in \mathcal{P}$ and $i \in Ag$.
- Kripke structure: $\mathcal{M} = (\mathcal{W}, \mathcal{R}_1, \ldots R_n, \rho)$ is a structure where: $R_i \subseteq (\mathcal{W} \times \mathcal{W})$ and $\rho : \mathcal{W} \to 2^{\mathcal{P}}$.
- Semantics:

  $\mathcal{M}, w \models p$ iff $p \in \rho(w)$

  $\mathcal{M}, w \models \Box_i\, \varphi$ iff for every $u \in \mathcal{W}$ if $(w, u) \in \mathcal{R}_i$ then $\mathcal{M}, u \models \varphi$.

Beyond fixed agent set

- In ML^n the agent set is fixed beforehand in the syntax.

Beyond fixed agent set

- In ML^n the agent set is fixed beforehand in the syntax.
- In many settings, index set cannot be fixed and changes dynamically.
    - Multi-process systems, Large games, Client-server systems

Beyond fixed agent set

- In ML^n the agent set is fixed beforehand in the syntax.
- In many settings, index set cannot be fixed and changes dynamically.
  - Multi-process systems, Large games, Client-server systems
- Agents change not just from one model to other, but also from one world to the other.

Beyond fixed agent set

- In ML^n the agent set is fixed beforehand in the syntax.
- In many settings, index set cannot be fixed and changes dynamically.
  - Multi-process systems, Large games, Client-server systems
- Agents change not just from one model to other, but also from one world to the other.

Examples

- $\forall x \Diamond_x p \;\wedge\; \exists y \Box_y p$

Beyond fixed agent set

- In ML^n the agent set is fixed beforehand in the syntax.
- In many settings, index set cannot be fixed and changes dynamically.
  - Multi-process systems, Large games, Client-server systems
- Agents change not just from one model to other, but also from one world to the other.

Examples

- $\forall x \Diamond_x p \ \wedge \ \exists y \Box_y p$
- $\forall x \forall y \ (\neg \Box_x \Box_y p \wedge \Box_x (\exists y \Diamond_y \neg p))$

Term modal Logic

- Term Modal Logic (TML) was introduced by Fitting, Thalmann and Voronkov [2001].

Term modal Logic

- Term Modal Logic (TML) was introduced by Fitting, Thalmann and Voronkov [2001].
- In TML, modalities are indexed by terms and these terms can be quantified.

Term modal Logic

- Term Modal Logic (TML) was introduced by Fitting, Thalmann and Voronkov [2001].
- In TML, modalities are indexed by terms and these terms can be quantified.

Examples

- Every process of that satisfies property A can possibly change the system state where it synchronises with some process that satisfies property B.
  $\forall x\big(A(x) \supset \Diamond_x(\exists y\ B(y) \wedge Syn(x,y))\big)$.

Term modal Logic

- Term Modal Logic (TML) was introduced by Fitting, Thalmann and Voronkov [2001].
- In TML, modalities are indexed by terms and these terms can be quantified.

Examples

- Every process of that satisfies property A can possibly change the system state where it synchronises with some process that satisfies property B.
  $\forall x\big(A(x) \supset \Diamond_x(\exists y\ B(y) \wedge Syn(x,y)\big)\big)$.
- For every client there is some server which will handle the request.
  $\forall x\big(C(x) \supset \exists y\ \big(S(y) \wedge \Box_y(\neg C(x)\big)\big)\big)$

Syntax

Given a set countable of propositions $\mathcal{P}$ and a countable set of variables $\mathcal{V}$ the syntax of term modal logic is defined by

$$\varphi \ := \ p \mid x = y \mid \neg\varphi \mid (\varphi \land \varphi) \mid \exists x \ \varphi \mid \Diamond_x \ \varphi$$

where $p \in \mathcal{P}$ and $x, y \in \mathcal{V}$.

Syntax

Given a set countable of propositions $\mathcal{P}$ and a countable set of variables $\mathcal{V}$ the syntax of term modal logic is defined by

$$\varphi \ := \ p \mid x = y \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \exists x \ \varphi \mid \Diamond_x \ \varphi$$

where $p \in \mathcal{P}$ and $x, y \in \mathcal{V}$.

In this talk

- Restrict to Propositional Term modal logic (PTML):
  In general, TML allows predicates $P(\overline{x})$ which allows to talk about properties of agents.

Syntax

Given a set countable of propositions $\mathcal{P}$ and a countable set of variables $\mathcal{V}$ the syntax of term modal logic is defined by

$$\varphi \;:=\; p \mid x = y \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \exists x\, \varphi \mid \Diamond_x\, \varphi$$

where $p \in \mathcal{P}$ and $x, y \in \mathcal{V}$.

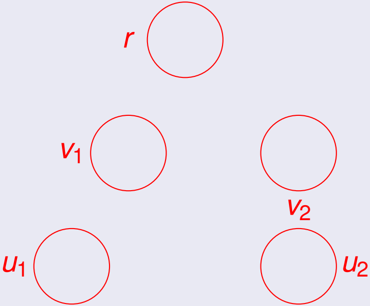
In this talk

- Restrict to Propositional Term modal logic (PTML):
  In general, TML allows predicates $P(\overline{x})$ which allows to talk about properties of agents.
- No constants, No function symbols

Structure

A TML model is given by $\mathcal{M} = (\mathcal{W},$

– $\mathcal{W}$ is non-empty set of worlds



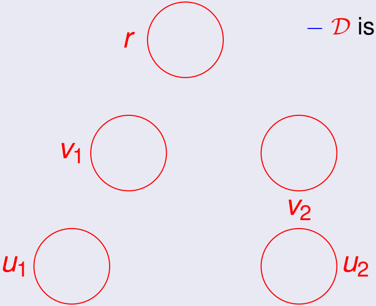
Structure

A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D},$

$\mathcal{D} = \{a, b, c, d, e\}$

– $\mathcal{W}$ is non-empty set of worlds

– $\mathcal{D}$ is non-empty set of potential agents



Structure

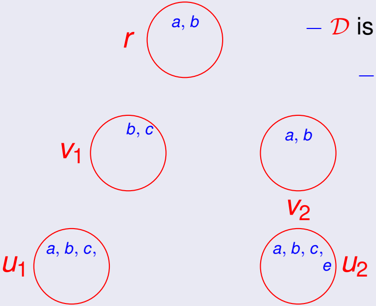
A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta,$

$\mathcal{D} = \{a, b, c, d, e\}$

– $\mathcal{W}$ is non-empty set of worlds

– $\mathcal{D}$ is non-empty set of potential agents

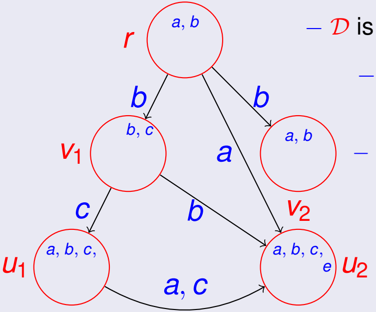
– $\delta : \mathcal{W} \mapsto 2^{\mathcal{D}}$ assigns to each w, a *non-empty* local agent set



Structure

A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R},$

$\mathcal{D} = \{a, b, c, d, e\}$



– $\mathcal{W}$ is non-empty set of worlds

– $\mathcal{D}$ is non-empty set of potential agents

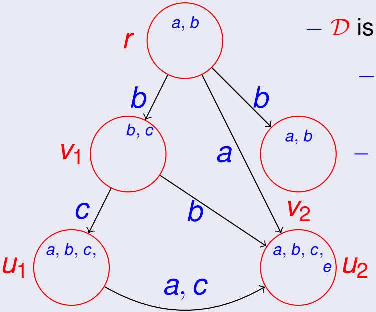
– $\delta : \mathcal{W} \mapsto 2^{\mathcal{D}}$ assigns to each w, a *non-empty* local agent set

– $\mathcal{R} \subseteq (\mathcal{W} \times \mathcal{D} \times \mathcal{W})$ is the accessiblity relation if $(w, i, w') \in \mathcal{R}$ then $i \in \delta(w)$

Structure

A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R},$

$\mathcal{D} = \{a, b, c, d, e\}$



– $\mathcal{W}$ is non-empty set of worlds

– $\mathcal{D}$ is non-empty set of potential agents

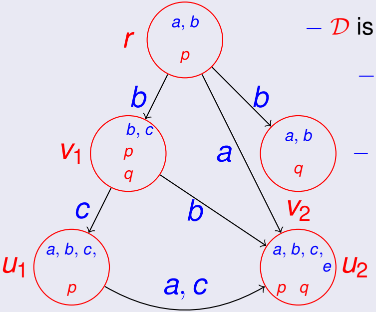
– $\delta : \mathcal{W} \mapsto 2^{\mathcal{D}}$ assigns to each w, a *non-empty* local agent set

– $\mathcal{R} \subseteq (\mathcal{W} \times \mathcal{D} \times \mathcal{W})$ is the accessiblity relation if $(w, i, w') \in \mathcal{R}$ then $i \in \delta(w)$

Structure

A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R}, \rho)$

$\mathcal{D} = \{a, b, c, d, e\}$



– $\mathcal{W}$ is non-empty set of worlds

– $\mathcal{D}$ is non-empty set of potential agents

– $\delta : \mathcal{W} \mapsto 2^{\mathcal{D}}$ assigns to each w, a *non-empty* local agent set

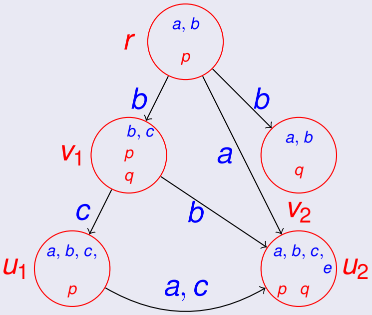
– $\mathcal{R} \subseteq (\mathcal{W} \times \mathcal{D} \times \mathcal{W})$ is the accessiblity relation if $(w, i, w') \in \mathcal{R}$ then $i \in \delta(w)$

– $\rho : (\mathcal{W}) \mapsto 2^{\mathcal{P}}$

Semantics

A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R}, \rho)$

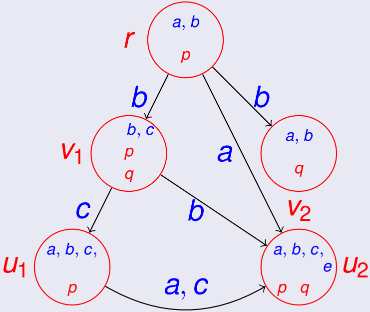
$\mathcal{D} = \{a, b, c, d, e\}$



Semantics

A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R}, \rho)$

$\mathcal{D} = \{a, b, c, d, e\}$

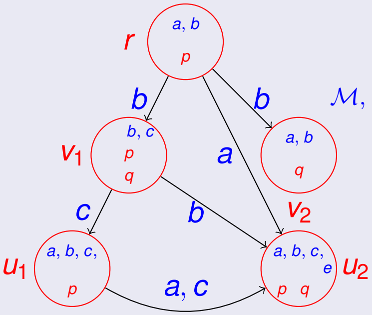


Interpretation: $\sigma : \mathcal{V} \rightarrow \mathcal{D}$

Semantics

A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R}, \rho)$

$\mathcal{D} = \{a, b, c, d, e\}$



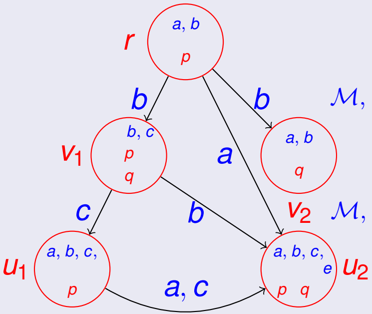
Interpretation: $\sigma : \mathcal{V} \to \mathcal{D}$

$\mathcal{M}, w, \sigma \models \Box_x\, \varphi$ if for all $(w, \sigma(x), u) \in \mathcal{R}$ we have $\mathcal{M}, u, \sigma \models \varphi$

Semantics

A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R}, \rho)$

$\mathcal{D} = \{a, b, c, d, e\}$



Interpretation: $\sigma : \mathcal{V} \to \mathcal{D}$

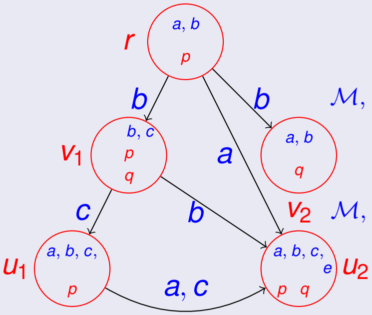
$\mathcal{M}, w, \sigma \models \Box_x \varphi$ if for all $(w, \sigma(x), u) \in \mathcal{R}$ we have $\mathcal{M}, u, \sigma \models \varphi$

$\mathcal{M}, w, \sigma \models \exists x\ \varphi$ if there is some $i \in \delta(w)$ such that $\mathcal{M}, w, \sigma[x \mapsto i] \models \varphi$

Semantics

A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R}, \rho)$

$\mathcal{D} = \{a, b, c, d, e\}$



Interpretation: $\sigma : \mathcal{V} \to \mathcal{D}$

$\mathcal{M}, w, \sigma \models \Box_x \varphi$ if for all $(w, \sigma(x), u) \in \mathcal{R}$ we have $\mathcal{M}, u, \sigma \models \varphi$

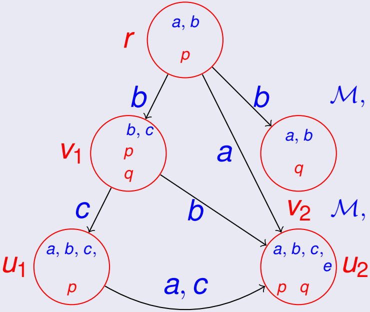
$\mathcal{M}, w, \sigma \models \exists x\ \varphi$ if there is some $i \in \delta(w)$ such that $\mathcal{M}, w, \sigma[x \mapsto i] \models \varphi$

$\mathcal{M}, r \models \forall x \Box_x \exists y\ \Box_y \neg q$

Semantics

A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R}, \rho)$

$\mathcal{D} = \{a, b, c, d, e\}$



Interpretation: $\sigma : \mathcal{V} \to \mathcal{D}$

$\mathcal{M}, w, \sigma \models \Box_x \varphi$ if for all $(w, \sigma(x), u) \in \mathcal{R}$ we have $\mathcal{M}, u, \sigma \models \varphi$

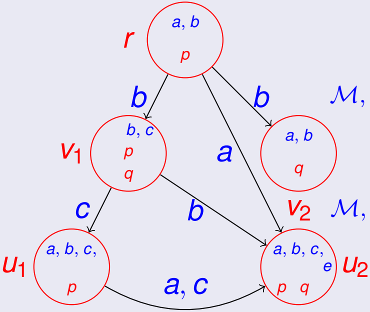
$\mathcal{M}, w, \sigma \models \exists x \ \varphi$ if there is some $i \in \delta(w)$ such that $\mathcal{M}, w, \sigma[x \mapsto i] \models \varphi$

$\mathcal{M}, r \models \forall x \forall y \ \Box_x \Box_y p$

Semantics

A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R}, \rho)$

$\mathcal{D} = \{a, b, c, d, e\}$



Interpretation: $\sigma : \mathcal{V} \to \mathcal{D}$

$\mathcal{M}, w, \sigma \models \Box_x \varphi$ if for all $(w, \sigma(x), u) \in \mathcal{R}$ we have $\mathcal{M}, u, \sigma \models \varphi$

$\mathcal{M}, w, \sigma \models \exists x \varphi$ if there is some $i \in \delta(w)$ such that $\mathcal{M}, w, \sigma[x \mapsto i] \models \varphi$

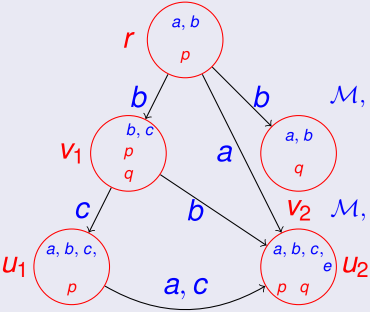
$\mathcal{M}, r \models \forall x \forall y \Box_x \Box_y p$

Evaluate $\Box_b \Box_a p$ at r

Semantics

A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R}, \rho)$

$\mathcal{D} = \{a, b, c, d, e\}$



Interpretation: $\sigma : \mathcal{V} \to \mathcal{D}$

$\mathcal{M}, w, \sigma \models \Box_x \varphi$ if for all $(w, \sigma(x), u) \in \mathcal{R}$ we have $\mathcal{M}, u, \sigma \models \varphi$

$\mathcal{M}, w, \sigma \models \exists x\, \varphi$ if there is some $i \in \delta(w)$ such that $\mathcal{M}, w, \sigma[x \mapsto i] \models \varphi$

$\mathcal{M}, r \models \forall x \forall y\, \Box_x \Box_y p$

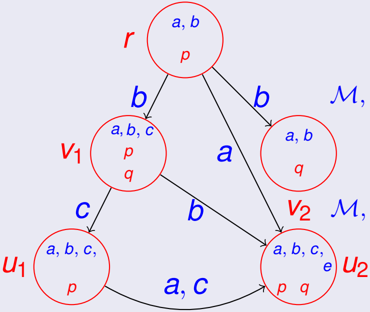
Evaluate $\Box_b \Box_a p$ at r

Evaluate $\Box_a p$ at v_1

Semantics

A TML model is given by $\mathcal{M} = (\mathcal{W}, \mathcal{D}, \delta, \mathcal{R}, \rho)$

$\mathcal{D} = \{a, b, c, d, e\}$ Monotonicty: if $(w, i, w') \in \mathcal{R}$ then $\delta(w) \subseteq \delta(w')$



Interpretation: $\sigma : \mathcal{V} \to \mathcal{D}$

$\mathcal{M}, w, \sigma \models \Box_x \varphi$ if for all $(w, \sigma(x), u) \in \mathcal{R}$ we have $\mathcal{M}, u, \sigma \models \varphi$

$\mathcal{M}, w, \sigma \models \exists x\ \varphi$ if there is some $i \in \delta(w)$ such that $\mathcal{M}, w, \sigma[x \mapsto i] \models \varphi$

$\mathcal{M}, r \models \forall x \forall y\ \Box_x \Box_y p$

Evaluate $\Box_b \Box_a p$ at r

Evaluate $\Box_a p$ at v_1

Satisfiability problem

Theorem

Satisfiability problem for TML when atoms are restricted to propositions is undecidable.

Satisfiability problem

Theorem

Satisfiability problem for TML when atoms are restricted to propositions is undecidable.

Proof sketch.

Reduction from FO satisfiability problem.

Satisfiability problem

Theorem

Satisfiability problem for TML when atoms are restricted to propositions is undecidable.

Proof sketch.

Reduction from FO satisfiability problem.

- $\mathrm{Tr}_1(Q(x_1, \ldots, x_n)) \rightsquigarrow \Diamond_{x_1} \Diamond_{x_2} \ldots \Diamond_{x_n} q$

Satisfiability problem

Theorem

Satisfiability problem for TML when atoms are restricted to propositions is undecidable.

Proof sketch.

Reduction from FO satisfiability problem.

- $\mathrm{Tr}_1(Q(x_1, \ldots, x_n)) \rightsquigarrow \Diamond_{x_1} \Diamond_{x_2} \ldots \Diamond_{x_n} q$
- $\mathrm{Tr}_1(\neg\varphi) \rightsquigarrow \neg\, \mathrm{Tr}_1(\varphi)$
- $\mathrm{Tr}_1(\varphi \wedge \psi) \rightsquigarrow \mathrm{Tr}_1(\varphi) \wedge \mathrm{Tr}_1(\psi)$

Satisfiability problem

Theorem

Satisfiability problem for TML when atoms are restricted to propositions is undecidable.

Proof sketch.

Reduction from FO satisfiability problem.

- $\mathrm{Tr}_1(Q(x_1, \ldots, x_n)) \rightsquigarrow \Diamond_{x_1} \Diamond_{x_2} \ldots \Diamond_{x_n} q$
- $\mathrm{Tr}_1(\neg\varphi) \rightsquigarrow \neg\ \mathrm{Tr}_1(\varphi)$
- $\mathrm{Tr}_1(\varphi \wedge \psi) \rightsquigarrow \mathrm{Tr}_1(\varphi) \wedge \mathrm{Tr}_1(\psi)$
- $\mathrm{Tr}_1(\exists x\ \varphi) \rightsquigarrow \exists x\ \mathrm{Tr}_1(\varphi)$

Satisfiability problem

Theorem

Satisfiability problem for TML when atoms are restricted to propositions is undecidable.

Proof sketch.

Reduction from FO satisfiability problem.

- $Tr_1(Q(x_1, \ldots, x_n)) \rightsquigarrow \Diamond_{x_1} \Diamond_{x_2} \ldots \Diamond_{x_n} q$
- $Tr_1(\neg\varphi) \rightsquigarrow \neg\, Tr_1(\varphi)$
- $Tr_1(\varphi \wedge \psi) \rightsquigarrow Tr_1(\varphi) \wedge Tr_1(\psi)$
- $Tr_1(\exists x\ \varphi) \rightsquigarrow \exists x\ Tr_1(\varphi)$

Any $\varphi \in$ FO is satisfiable iff $Tr_1(\varphi)$ is satisfiable.

$TML^=$

With only $=$ in the syntax, $\top := \forall x\,(x = x)$. Hence $TML^=$ is undecidable.

$TML^=$

With only $=$ in the syntax, $\top := \forall x\,(x = x)$. Hence $TML^=$ is undecidable.

Theorem (Trakhtenbrot type theorem)

For $TML^=$, the FinSat, UnSat and InfAx are mutually recursively inseparable [a].

[a] Sets X and Y are recursively inseparable if there is no Z such that $X \subseteq Z$ and $Z \cap Y = \emptyset$ where checking for membership in Z is decidable.

$TML^=$

With only $=$ in the syntax, $\top := \forall x\,(x = x)$. Hence $TML^=$ is undecidable.

Theorem (Trakhtenbrot type theorem)

For $TML^=$, the FinSat, UnSat and InfAx are mutually recursively inseparable [a]*.*

---

[a] Sets X and Y are recursively inseparable if there is no Z such that $X \subseteq Z$ and $Z \cap Y = \emptyset$ where checking for membership in Z is decidable.

Reduction via tiling

For any tiling instance T, we come up with a formula $\varphi_T \in TML^=$ such that

$TML^=$

With only $=$ in the syntax, $\top := \forall x\ (x = x)$. Hence $TML^=$ is undecidable.

Theorem (Trakhtenbrot type theorem)

For $TML^=$, the FinSat, UnSat and InfAx are mutually recursively inseparable [a].

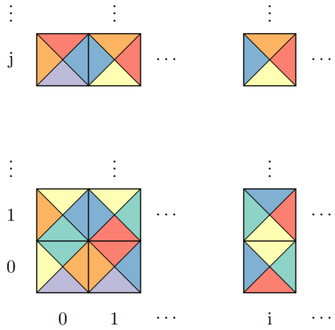
[a] Sets X and Y are recursively inseparable if there is no Z such that $X \subseteq Z$ and $Z \cap Y = \emptyset$ where checking for membership in Z is decidable.

Reduction via tiling

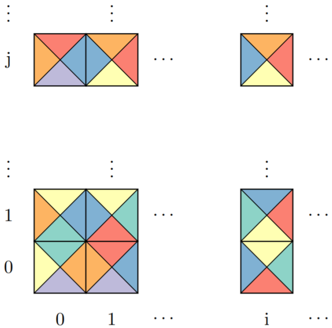
For any tiling instance T, we come up with a formula $\varphi_T \in TML^=$ such that

- $\varphi_T \in$ FinSat iff T has some periodic tiling.
- $\varphi_T \in$ InfAx iff T has only aperiodic tiling.
- $\varphi_T \in$ UnSat iff T has no tiling.

Tiling $\mathbb{N} \times \mathbb{N}$



Tiling $\mathbb{N} \times \mathbb{N}$



- We induce a discreet linear order with min over agents and use it to construct the grid.

Discreet linear order in TML$^=$

Encode $x < y$ as $\Diamond_x \Diamond_y \top$.

Discreet linear order in TML$^=$

Encode $x < y$ as $\Diamond_x \Diamond_y \top$.

φ_0	:=	$\exists x \; zero(x)$
φ_{ir}	:=	$\forall x \; \neg \Diamond_x \Diamond_x \top$
φ_{tot}	:=	$\forall x \forall y \; ((x \neq y) \supset (\Diamond_x \Diamond_y \top \vee \Diamond_y \Diamond_x \top))$
φ_{dis}	:=	$\forall x \; (last(x) \vee \exists y \; succ(x, y))$
φ_{trans}	:=	$\forall x \forall y \forall z \; (\Diamond_x \Diamond_y \top \wedge \Diamond_y \Diamond_z \top) \supset (\Diamond_x \Diamond_z \top)$
where,		
$zero(x)$	:=	$\forall y \; \neg \Diamond_y \Diamond_x \top$
$last(x)$	:=	$\forall y \; \neg \Diamond_x \Diamond_y \top$
$succ(x, y)$	:=	$\Diamond_x \Diamond_y \top \; \wedge \forall z \; ((\Diamond_z \Diamond_y \top) \supset (x = z \vee \Diamond_z \Diamond_x \top))$

TML$^=$

Define $\mathcal{O} = \{\varphi_0, \varphi_{ir}, \varphi_{tot}, \varphi_{dis}\}$ and $\hat{\mathcal{O}} = \bigwedge_{\varphi \in \mathcal{O}} \varphi$.

TML$^=$

Define $\mathcal{O} = \{\varphi_0, \varphi_{ir}, \varphi_{tot}, \varphi_{dis}\}$ and $\hat{\mathcal{O}} = \bigwedge_{\varphi \in \mathcal{O}} \varphi$.

Lemma

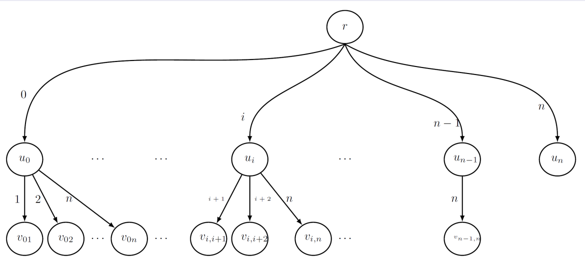
For every $\mathbb{N}' \subseteq \mathbb{N}$ (either finite or infinite) which is an initial segment of $\mathbb{N}$, there is some $\mathcal{M} = (\mathcal{W}, \mathbb{N}', \delta, \mathcal{R})$ and $r \in \mathcal{W}$ such that $\delta(r) = \mathbb{N}'$ and $\mathcal{M}, r \models \hat{\mathcal{O}}$.

TML$^=$

Define $\mathcal{O} = \{\varphi_0, \varphi_{ir}, \varphi_{tot}, \varphi_{dis}\}$ and $\hat{\mathcal{O}} = \bigwedge_{\varphi \in \mathcal{O}} \varphi$.

Lemma

For every $\mathbb{N}' \subseteq \mathbb{N}$ (either finite or infinite) which is an initial segment of $\mathbb{N}$, there is some $\mathcal{M} = (\mathcal{W}, \mathbb{N}', \delta, \mathcal{R})$ and $r \in \mathcal{W}$ such that $\delta(r) = \mathbb{N}'$ and $\mathcal{M}, r \models \hat{\mathcal{O}}$.



Discreet linear order in TML$^=$

Lemma

For any model $\mathcal{M}$ and $r \in \mathcal{W}$, suppose $\mathcal{M}, r \models \hat{\mathcal{O}}$ then let $\mathbb{N}' = [0, \ldots, n-1]$ if $\delta(r)$ is a finite set of size n and $\mathbb{N}' = \mathbb{N}$ otherwise.

Discreet linear order in $TML^=$

Lemma

For any model $\mathcal{M}$ and $r \in \mathcal{W}$, suppose $\mathcal{M}, r \models \hat{\mathcal{O}}$ then let $\mathbb{N}' = [0, \ldots, n-1]$ if $\delta(r)$ is a finite set of size n and $\mathbb{N}' = \mathbb{N}$ otherwise. There exists a function $f : \mathbb{N}' \to \delta(r)$ such that for all $i, j \in \mathbb{N}'$ if $i < j$ then $\mathcal{M}, w \models \Diamond_{f(i)} \Diamond_{f(j)} \top$.

Discreet linear order in $TML^=$

Lemma

For any model $\mathcal{M}$ *and* $r \in \mathcal{W}$, *suppose* $\mathcal{M}, r \models \hat{\mathcal{O}}$ *then let* $\mathbb{N}' = [0, \ldots, n-1]$ *if* $\delta(r)$ *is a finite set of size* n *and* $\mathbb{N}' = \mathbb{N}$ *otherwise. There exists a function* $f : \mathbb{N}' \to \delta(r)$ *such that for all* $i, j \in \mathbb{N}'$ *if* $i < j$ *then* $\mathcal{M}, w \models \Diamond_{f(i)} \Diamond_{f(j)} \top$.

- Therefore, whenever $\mathcal{M}, r \models \hat{\mathcal{O}}$, wlog there is some initial fragment $\mathbb{N}'$ of $\mathbb{N}$ with $|\mathbb{N}'| = |\delta(r)|$ such that $\mathbb{N}' \subseteq \delta(r)$ and for all $i, j \in \mathbb{N}'$ if $i < j$ then $\mathcal{M}, w \models \Diamond_i \Diamond_j \top$.

Encoding tiling in $TML^=$

- Encode every tile t_i as a path of length i, given by

$$p_i ::= \bigwedge_{j<i} (\forall z \Box_z)^j (\exists z \Diamond_z \top) \land (\forall z \Box_z)^i (\forall z \Box_z \bot)$$

Encoding tiling in TML$^=$

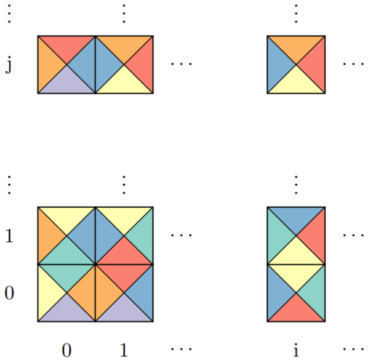
- Encode every tile t_i as a path of length i, given by

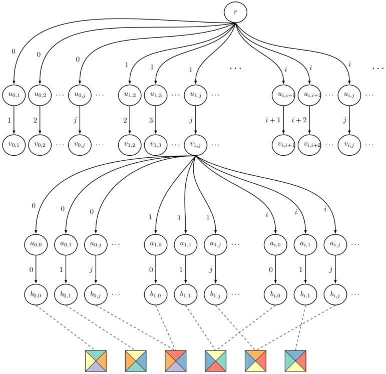
$$p_i ::= \bigwedge_{j<i} (\forall z \Box_z)^j (\exists z \Diamond_z \top) \wedge (\forall z \Box_z)^i (\forall z \Box_z \bot)$$

- (j, k) has tile t_i is encoded as $\Box_j \Box_k p_i$.

$\varphi tile$	$:=$	$\forall z_1 \forall z_2 \forall x \forall y\ \Box_{z_1} \Box_{z_2} ((\Diamond_x \Diamond_y \top) \wedge (\Box_x \Box_y \bigvee_{t_i \in X} p_i))$
$\varphi init$	$:=$	$\forall x\ zero(x) \Rightarrow \forall z_1 \forall z_2 (\Box_{z_1} \Box_{z_2} \Diamond_x \Diamond_x p_0)$
φhor	$:=$	$\forall x \forall y \forall z\ succ(x, y) \Rightarrow$ $(\forall z_1 \forall z_2 \Box_{z_1} \Box_{z_2} (\bigvee_{r_{t_i} = \ell_{t_j}} (\Box_x \Box_z (p_i) \wedge \Box_y \Box_z (p_j)))$
φver	$:=$	$\forall x \forall y \forall z\ succ(x, y) \Rightarrow$ $\forall z_1 \forall z_2 (\Box_{z_1} \Box_{z_2} (\bigvee_{u_{t_i} = d_{t_j}} (\Box_z \Box_x (p_i) \wedge \Box_z \Box_y (p_j)))$

Tiling in TML$^=$





Decidable fragments

- One approach to get decidable fragments:
  Embed TML into First order modal logic (FOML) and look at the decidable fragments of FOML in literature.

Decidable fragments

- One approach to get decidable fragments: Embed TML into First order modal logic (FOML) and look at the decidable fragments of FOML in literature.

Translating TML to FOML

- $\mathrm{Tr}_2(p) \rightsquigarrow p$

Decidable fragments

- One approach to get decidable fragments: Embed TML into First order modal logic (FOML) and look at the decidable fragments of FOML in literature.

Translating TML to FOML

- $\text{Tr}_2(p) \leadsto p$
- $\text{Tr}_2(\neg\varphi) \leadsto \neg\, \text{Tr}_2(\varphi)$
- $\text{Tr}_2(\varphi \wedge \psi) \leadsto \text{Tr}_2(\varphi) \wedge \text{Tr}_2(\psi)$

Decidable fragments

- One approach to get decidable fragments: Embed TML into First order modal logic (FOML) and look at the decidable fragments of FOML in literature.

Translating TML to FOML

- $Tr_2(p) \rightsquigarrow p$
- $Tr_2(\neg\varphi) \rightsquigarrow \neg\ Tr_2(\varphi)$
- $Tr_2(\varphi \wedge \psi) \rightsquigarrow Tr_2(\varphi) \wedge Tr_2(\psi)$
- $Tr_2(\exists x\ \varphi) \rightsquigarrow \exists x\ Tr_2(\varphi)$

Decidable fragments

- One approach to get decidable fragments: Embed TML into First order modal logic (FOML) and look at the decidable fragments of FOML in literature.

Translating TML to FOML

- $Tr_2(p) \rightsquigarrow p$
- $Tr_2(\neg\varphi) \rightsquigarrow \neg\ Tr_2(\varphi)$
- $Tr_2(\varphi \wedge \psi) \rightsquigarrow Tr_2(\varphi) \wedge Tr_2(\psi)$
- $Tr_2(\exists x\ \varphi) \rightsquigarrow \exists x\ Tr_2(\varphi)$
- $Tr_2(\Box_x\ \varphi) \rightsquigarrow \Box\ (E(x) \supset Tr_2(\varphi))$
  where E is a new unary predicate.

Decidable fragments

- One approach to get decidable fragments: Embed TML into First order modal logic (FOML) and look at the decidable fragments of FOML in literature.

Translating TML to FOML

- $Tr_2(p) \leadsto p$
- $Tr_2(\neg\varphi) \leadsto \neg\ Tr_2(\varphi)$
- $Tr_2(\varphi \wedge \psi) \leadsto Tr_2(\varphi) \wedge Tr_2(\psi)$
- $Tr_2(\exists x\ \varphi) \leadsto \exists x\ Tr_2(\varphi)$
- $Tr_2(\Box_x\ \varphi) \leadsto \Box\ (E(x) \supset Tr_2(\varphi))$
  where E is a new unary predicate.

Any $\varphi \in$ TML is satisfiable iff the corresponding FOML formula $Tr_2(\varphi)$ is satisfiable.

Decidable fragments via FOML

Monodic fragment (Wolter and Zakharyaschev)

- Generalization of 1 variable fragment.

Decidable fragments via FOML

Monodic fragment (Wolter and Zakharyaschev)

- Generalization of 1 variable fragment.
- $\varphi \in$ FOML is monodic if for any subformula $\Delta\alpha$ we have $|FV(\alpha)| \leq 1$ where $\Delta = \{\Box, \Diamond\}$.

Decidable fragments via FOML

Monodic fragment (Wolter and Zakharyaschev)

- Generalization of 1 variable fragment.
- $\varphi \in$ FOML is monodic if for any subformula $\Delta\alpha$ we have $|FV(\alpha)| \leq 1$ where $\Delta = \{\Box, \Diamond\}$.
- $\psi \in$ TML is monodic if for any subformula $\Delta_x\alpha$ we have $FV(\alpha) \subseteq \{x\}$ where $\Delta = \{\Box, \Diamond\}$.

Decidable fragments via FOML

Monodic fragment (Wolter and Zakharyaschev)

- Generalization of 1 variable fragment.
- $\varphi \in$ FOML is monodic if for any subformula $\Delta\alpha$ we have $|\mathsf{FV}(\alpha)| \leq 1$ where $\Delta = \{\Box, \Diamond\}$.
- $\psi \in$ TML is monodic if for any subformula $\Delta_x\alpha$ we have $\mathsf{FV}(\alpha) \subseteq \{x\}$ where $\Delta = \{\Box, \Diamond\}$.
- Monodic restriction of FOML over any decidable fragment of FO (Ex: Unary predicates) continues to be decidable.

Decidable fragments via FOML

Monodic fragment (Wolter and Zakharyaschev)

- Generalization of 1 variable fragment.
- $\varphi \in$ FOML is monodic if for any subformula $\Delta\alpha$ we have $|FV(\alpha)| \leq 1$ where $\Delta = \{\Box, \Diamond\}$.
- $\psi \in$ TML is monodic if for any subformula $\Delta_x\alpha$ we have $FV(\alpha) \subseteq \{x\}$ where $\Delta = \{\Box, \Diamond\}$.
- Monodic restriction of FOML over any decidable fragment of FO (Ex: Unary predicates) continues to be decidable.
- Translation preserves monodicity.

Decidable fragments via FOML

Monodic fragment (Wolter and Zakharyaschev)

- Generalization of 1 variable fragment.
- $\varphi \in$ FOML is monodic if for any subformula $\Delta\alpha$ we have $|FV(\alpha)| \leq 1$ where $\Delta = \{\Box, \Diamond\}$.
- $\psi \in$ TML is monodic if for any subformula $\Delta_x\alpha$ we have $FV(\alpha) \subseteq \{x\}$ where $\Delta = \{\Box, \Diamond\}$.
- Monodic restriction of FOML over any decidable fragment of FO (Ex: Unary predicates) continues to be decidable.
- Translation preserves monodicity.
- Monodic fragment of TML over unary predicates is decidable.

Decidable fragments via FOML

Bundled fragment (P., Ramanujam, Wang)

- Quantifiers and modalities occur in together: $\forall x \Box \alpha$ and $\exists x \Box \alpha$ ($\exists x \Diamond \alpha$ and $\forall x \Diamond \alpha$ dually)

Decidable fragments via FOML

Bundled fragment (P., Ramanujam, Wang)

- Quantifiers and modalities occur in together: $\forall x \Box \alpha$ and $\exists x \Box \alpha$ ($\exists x \Diamond \alpha$ and $\forall x \Diamond \alpha$ dually)
- No restriction on the arity of predicates.

Decidable fragments via FOML

Bundled fragment (P., Ramanujam, Wang)

- Quantifiers and modalities occur in together:
  $\forall x \Box \alpha$ and $\exists x \Box \alpha$ ($\exists x \Diamond \alpha$ and $\forall x \Diamond \alpha$ dually)
- No restriction on the arity of predicates.
- Useful to study Know what, why, how etc [Wang 2017]

Decidable fragments via FOML

Bundled fragment (P., Ramanujam, Wang)

- Quantifiers and modalities occur in together: $\forall x \Box \alpha$ and $\exists x \Box \alpha$ ($\exists x \Diamond \alpha$ and $\forall x \Diamond \alpha$ dually)
- No restriction on the arity of predicates.
- Useful to study Know what, why, how etc [Wang 2017]
- Bundled fragment of First order modal logic is PSPACE-complete.

Decidable fragments via FOML

Bundled fragment (P., Ramanujam, Wang)

- Quantifiers and modalities occur in together: $\forall x \Box \alpha$ and $\exists x \Box \alpha$ ($\exists x \Diamond \alpha$ and $\forall x \Diamond \alpha$ dually)
- No restriction on the arity of predicates.
- Useful to study Know what, why, how etc [Wang 2017]
- Bundled fragment of First order modal logic is PSPACE-complete.
- Bundled fragment of TML: $\forall x \Box_x \alpha$ and $\exists x \Box_x \alpha$

Decidable fragments via FOML

Bundled fragment (P., Ramanujam, Wang)

- Quantifiers and modalities occur in together: $\forall x \Box \alpha$ and $\exists x \Box \alpha$ ($\exists x \Diamond \alpha$ and $\forall x \Diamond \alpha$ dually)
- No restriction on the arity of predicates.
- Useful to study Know what, why, how etc [Wang 2017]
- Bundled fragment of First order modal logic is PSPACE-complete.
- Bundled fragment of TML: $\forall x \Box_x \alpha$ and $\exists x \Box_x \alpha$
- Translation preserves bundled formulas.

TML on 2-variables

- 3 variable TML is undecidable (with just $\top, \bot$ as atoms)

TML on 2-variables

- 3 variable TML is undecidable (with just $\top, \bot$ as atoms)
- 1 variable TML is decidable (1 variable formulas are always monodic)

TML on 2-variables

- 3 variable TML is undecidable (with just $\top, \bot$ as atoms)
- 1 variable TML is decidable (1 variable formulas are always monodic)
- Rest of the talk: 2 variableTML is decidable

TML on 2-variables

- 3 variable TML is undecidable (with just $\top, \bot$ as atoms)
- 1 variable TML is decidable (1 variable formulas are always monodic)
- Rest of the talk: 2 variableTML is decidable
- Contrast with 2 variable FOML which is undecidable

TML on 2-variables

- 3 variable TML is undecidable (with just $\top, \bot$ as atoms)
- 1 variable TML is decidable (1 variable formulas are always monodic)
- Rest of the talk: 2 variableTML is decidable
- Contrast with 2 variable FOML which is undecidable

- Quoting Wolter and Zakharyaschev:

  All undecidability proofs for fragments of modal predicate logics exploit formulas of the form $\Box\psi(x, y)$ in which the necessity operator applies to subformulas of more than one free variable; in fact, such formulas play an essential role in the reduction of undecidable problems to those fragments . . .

TML on 2-variables

- 3 variable TML is undecidable (with just $\top, \bot$ as atoms)
- 1 variable TML is decidable (1 variable formulas are always monodic)
- Rest of the talk: 2 variableTML is decidable
- Contrast with 2 variable FOML which is undecidable

- Quoting Wolter and Zakharyaschev:

  All undecidability proofs for fragments of modal predicate logics exploit formulas of the form $\Box\psi(x, y)$ in which the necessity operator applies to subformulas of more than one free variable; in fact, such formulas play an essential role in the reduction of undecidable problems to those fragments . . .

- Not expressible in TML: (we only have $\Box_x\psi(x, y)$ or $\Box_y\psi(x, y)$. Crucial to get decidability.

PTML^2 decidability

Proof strategy

- Normal form for PTML^2: Combination of Modal normal form and FO^2 normal form

$PTML^2$ decidability

Proof strategy

- Normal form for $PTML^2$: Combination of Modal normal form and FO^2 normal form
- Bounded model property: Induction over FO^2 bounded model construction

FO^2 decidability

Scott normal form

$\forall x \forall y \, \varphi \wedge \bigwedge_i \forall x \exists y \, \psi_i$ where φ, ψ_i are quantifier free.

FO^2 decidability

Scott normal form

$\forall x \forall y\ \varphi \wedge \bigwedge_i \forall x \exists y\ \psi_i$ where φ, ψ_i are quantifier free.

Every $\alpha \in$ FO^2 has an equi-satisfiable formula in the normal form.

FO^2 decidability

Scott normal form

$\forall x \forall y\ \varphi \wedge \bigwedge_i \forall x \exists y\ \psi_i$ where φ, ψ_i are quantifier free.

Every $\alpha \in FO^2$ has an equi-satisfiable formula in the normal form.

FO types

For a given FO structure $\mathfrak{A} = (\mathcal{D}, \rho)$

- For all $c, d \in \mathcal{D}$
  $\text{2-type}(c, d) = \{(\neg)P(x, y) \mid \mathfrak{A} \models (\neg)P(c, d)\}$

FO^2 decidability

Scott normal form

$\forall x \forall y\ \varphi \wedge \bigwedge_i \forall x \exists y\ \psi_i$ where φ, ψ_i are quantifier free.

Every $\alpha \in$ FO^2 has an equi-satisfiable formula in the normal form.

FO types

For a given FO structure $\mathfrak{A} = (\mathcal{D}, \rho)$

- For all $c, d \in \mathcal{D}$
  $\text{2-type}(c, d) = \{(\neg)P(x, y) \mid \mathfrak{A} \models (\neg)P(c, d)\}$
- $\text{1-type}(c) = (\Lambda_1; \Lambda_2)$ where
  $\Lambda_1 = \text{2-type}(c, c)$ and $\Lambda_2 = \{\text{2-type}(c, d) \mid d \in \mathcal{D}\}$.

FO^2 decidability

Scott normal form

$\forall x \forall y\ \varphi \wedge \bigwedge_i \forall x \exists y\ \psi_i$ where φ, ψ_i are quantifier free.

Every $\alpha \in FO^2$ has an equi-satisfiable formula in the normal form.

FO types

For a given FO structure $\mathfrak{A} = (\mathcal{D}, \rho)$

- For all $c, d \in \mathcal{D}$
  $\text{2-type}(c, d) = \{(\neg)P(x, y) \mid \mathfrak{A} \models (\neg)P(c, d)\}$
- $\text{1-type}(c) = (\Lambda_1; \Lambda_2)$ where
  $\Lambda_1 = \text{2-type}(c, c)$ and $\Lambda_2 = \{\text{2-type}(c, d) \mid d \in \mathcal{D}\}$.

With respect to a given formula, there are at most exponentially many distinct 1-types.

FO^2 decidability

Bounded model construction

- If $(\mathcal{D}, \rho) \models \forall x \forall y \, \varphi \;\wedge\; \bigwedge_{i=1}^{q} \forall x \exists y \, \psi_i$

FO^2 decidability

Bounded model construction

- If $(\mathcal{D}, \rho) \models \forall x \forall y\ \varphi\ \wedge\ \bigwedge_{i=1}^{q} \forall x \exists y\ \psi_i$
- Let $\text{1-type}(\mathcal{D}) = \{\text{1-type}(c) \mid c \in \mathcal{D}\}$

FO^2 decidability

Bounded model construction

- If $(\mathcal{D}, \rho) \models \forall x \forall y\ \varphi \ \wedge \ \bigwedge_{i=1}^{q} \forall x \exists y\ \psi_i$
- Let 1-type$(\mathcal{D}) = \{$1-type$(c) \mid c \in \mathcal{D}\}$
- Define $\mathcal{D}' =$ 1-type$(\mathfrak{A}) \times [1 \ldots q] \times \{0, 1, 2\}$.

FO^2 decidability

Bounded model construction

- If $(\mathcal{D}, \rho) \models \forall x \forall y\ \varphi\ \wedge\ \bigwedge_{i=1}^{q} \forall x \exists y\ \psi_i$
- Let $\text{1-type}(\mathcal{D}) = \{\text{1-type}(c) \mid c \in \mathcal{D}\}$
- Define $\mathcal{D}' = \text{1-type}(\mathfrak{A}) \times [1 \ldots q] \times \{0, 1, 2\}$.
- For every $\Lambda \in \text{1-type}(\mathfrak{A})$ let $a_\Lambda \in \mathcal{D}$ such that $\text{1-type}(a_\Lambda) = \Lambda$.

FO^2 decidability

Bounded model construction

- If $(\mathcal{D}, \rho) \models \forall x \forall y\ \varphi\ \wedge\ \bigwedge_{i=1}^{q} \forall x \exists y\ \psi_i$
- Let $1\text{-type}(\mathcal{D}) = \{1\text{-type}(c) \mid c \in \mathcal{D}\}$
- Define $\mathcal{D}' = 1\text{-type}(\mathfrak{A}) \times [1 \ldots q] \times \{0, 1, 2\}$.
- For every $\Lambda \in 1\text{-type}(\mathfrak{A})$ let $a_\Lambda \in \mathcal{D}$ such that $1\text{-type}(a_\Lambda) = \Lambda$. For every such a_Λ let $b_1, \ldots b_q$ be the witness for $\psi_1 \ldots \psi_q$ and let $1\text{-type}(b_i) = \Pi_i$.

FO^2 decidability

Bounded model construction

- If $(\mathcal{D}, \rho) \models \forall x \forall y\ \varphi\ \wedge\ \bigwedge_{i=1}^{q} \forall x \exists y\ \psi_i$
- Let $\text{1-type}(\mathcal{D}) = \{\text{1-type}(c) \mid c \in \mathcal{D}\}$
- Define $\mathcal{D}' = \text{1-type}(\mathfrak{A}) \times [1 \ldots q] \times \{0, 1, 2\}$.
- For every $\Lambda \in \text{1-type}(\mathfrak{A})$ let $a_\Lambda \in \mathcal{D}$ such that $\text{1-type}(a_\Lambda) = \Lambda$. For every such a_Λ let $b_1, \ldots b_q$ be the witness for $\psi_1 \ldots \psi_q$ and let $\text{1-type}(b_i) = \Pi_i$.
    - $\text{2-type}[(\Lambda, e, f), (\Pi_i, i, f')] = \text{2-type}(a_\Lambda, b_i)$ where $f' = f + 1 \mod 3$

FO^2 decidability

Bounded model construction

- If $(\mathcal{D}, \rho) \models \forall x \forall y\ \varphi\ \wedge\ \bigwedge_{i=1}^{q} \forall x \exists y\ \psi_i$
- Let $\text{1-type}(\mathcal{D}) = \{\text{1-type}(c) \mid c \in \mathcal{D}\}$
- Define $\mathcal{D}' = \text{1-type}(\mathfrak{A}) \times [1 \ldots q] \times \{0, 1, 2\}$.
- For every $\Lambda \in \text{1-type}(\mathfrak{A})$ let $a_\Lambda \in \mathcal{D}$ such that $\text{1-type}(a_\Lambda) = \Lambda$. For every such a_Λ let $b_1, \ldots b_q$ be the witness for $\psi_1 \ldots \psi_q$ and let $\text{1-type}(b_i) = \Pi_i$.
    - $\text{2-type}[(\Lambda, e, f), (\Pi_i, i, f')] = \text{2-type}(a_\Lambda, b_i)$ where $f' = f + 1 \mod 3$
    - $\text{2-type}[(\Lambda, e, f), (\Pi, e', f')] = \text{2-type}(a_\Lambda, a_\Pi)$ if it is not defined in the previous step.

The type based model continues to satisfy

$$\forall x \forall y\ \varphi\ \wedge\ \bigwedge_{i=1}^{q} \forall x \exists y\ \psi_i$$

Normal form for PTML^2

- Fine normal form for modal logic: DNF where each clause is of the form: $(\bigwedge_i s_i \wedge \Box\alpha \wedge \bigwedge_j \Diamond\beta_j)$ where α, β_j are recursively in the normal form.

Normal form for $PTML^2$

- Fine normal form for modal logic: DNF where each clause is of the form: $(\bigwedge_i s_i \wedge \Box\alpha \wedge \bigwedge_j \Diamond\beta_j)$ where α, β_j are recursively in the normal form.
- Scott normal form for FO^2: $\forall x \forall y\, \varphi \wedge \bigwedge_i \forall x \exists y\, \psi_i$ where φ, ψ_i are quantifier free.

Normal form for PTML^2

- Fine normal form for modal logic: DNF where each clause is of the form: $(\bigwedge_i s_i \wedge \Box\alpha \wedge \bigwedge_j \Diamond\beta_j)$ where α, β_j are recursively in the normal form.
- Scott normal form for FO^2: $\forall x \forall y\ \varphi \wedge \bigwedge_i \forall x \exists y\ \psi_i$ where φ, ψ_i are quantifier free.
- Fine Scott normal form for PTML^2: DNF where each clause is of the form:

$$\bigwedge_{i \leq a} s_i \wedge \bigwedge_{z \in \{x,y\}} (\Box_z \alpha \wedge \bigwedge_{j \leq m_z} \Diamond_z \beta_j) \bigwedge_{z \in \{x,y\}} (\forall z\ \gamma \wedge \bigwedge_{k \leq n_z} \exists z\ \delta_k) \wedge \forall x \forall y\ \varphi \wedge \bigwedge_{l \leq b} \forall x \exists y\ \psi_l$$

where s_i are literals and all α, β_j are recursively in the normal form and $\gamma, \delta_k, \varphi, \psi_l$ do not have quantifiers at the outermost level.

PTML^2 type

For any tree model $\mathcal{M}$ rooted at r, (let $\Delta \in \{\Box, \Diamond\}$ and $z \in \{x, y\}$):

- For all $w \in \mathcal{W}$ and for all $c, d \in \delta(w)$, define $\textit{2-type}(w, c, d) = \{\Delta_z \psi(x, y) \mid \mathcal{M}, w \models \Delta_z \psi(c, d)\}$

PTML^2 type

For any tree model $\mathcal{M}$ rooted at r, (let $\Delta \in \{\Box, \Diamond\}$ and $z \in \{x, y\}$):

- For all $w \in \mathcal{W}$ and for all $c, d \in \delta(w)$, define $2\text{-}type(w, c, d) = \{\Delta_z \psi(x, y) \mid \mathcal{M}, w \models \Delta_z \psi(c, d)\}$
- If w is a non root node, (say $w' \xrightarrow{a} w$) then for all $c \in \delta(w)$ define $1\text{-type}(w, c) = (\Lambda_1; \Lambda_2; \Lambda_3)$

PTML^2 type

For any tree model $\mathcal{M}$ rooted at r, (let $\Delta \in \{\Box, \Diamond\}$ and $z \in \{x, y\}$):

- For all $w \in \mathcal{W}$ and for all $c, d \in \delta(w)$, define $\textit{2-type}(w, c, d) = \{\Delta_z\psi(x, y) \mid \mathcal{M}, w \models \Delta_z\psi(c, d)\}$
- If w is a non root node, (say $w' \xrightarrow{a} w$) then for all $c \in \delta(w)$ define $\text{1-type}(w, c) = (\Lambda_1; \Lambda_2; \Lambda_3)$
  $\Lambda_1 = \text{2-type}(w, c, c)$; $\Lambda_2 = \{\text{2-type}(w, c, d) \mid d \in \delta(w)\}$;

PTML^2 type

For any tree model $\mathcal{M}$ rooted at r, (let $\Delta \in \{\Box, \Diamond\}$ and $z \in \{x, y\}$):

- For all $w \in \mathcal{W}$ and for all $c, d \in \delta(w)$, define $2\text{-}type(w, c, d) = \{\Delta_z \psi(x, y) \mid \mathcal{M}, w \models \Delta_z \psi(c, d)\}$
- If w is a non root node, (say $w' \xrightarrow{a} w$) then for all $c \in \delta(w)$ define $1\text{-type}(w, c) = (\Lambda_1; \Lambda_2; \Lambda_3)$
  $\Lambda_1 = 2\text{-type}(w, c, c)$; $\Lambda_2 = \{2\text{-type}(w, c, d) \mid d \in \delta(w)\}$;
  $\Lambda_3 = 2\text{-type}(w, c, a)$

$PTML^2$ type

For any tree model $\mathcal{M}$ rooted at r, (let $\Delta \in \{\Box, \Diamond\}$ and $z \in \{x, y\}$):

- For all $w \in \mathcal{W}$ and for all $c, d \in \delta(w)$, define $2\text{-}type(w, c, d) = \{\Delta_z \psi(x, y) \mid \mathcal{M}, w \models \Delta_z \psi(c, d)\}$
- If w is a non root node, (say $w' \xrightarrow{a} w$) then for all $c \in \delta(w)$ define $1\text{-type}(w, c) = (\Lambda_1; \Lambda_2; \Lambda_3)$
  $\Lambda_1 = 2\text{-type}(w, c, c)$; $\Lambda_2 = \{2\text{-type}(w, c, d) \mid d \in \delta(w)\}$;
  $\Lambda_3 = 2\text{-type}(w, c, a)$
- For the root node r, for all $c \in \delta(r)$ define $1\text{-type}(w, c) = (\Lambda_1; \Lambda_2; \{\top\})$

Bounded model construction for PTML^2

Suppose $\mathcal{M}, r \models \theta$ where θ is in the Fine Scott normal form.

Bounded model construction for PTML^2

Suppose $\mathcal{M}, r \models \theta$ where θ is in the Fine Scott normal form.

$$\bigwedge_{i \leq a} s_i \wedge \bigwedge_{z \in \{x,y\}} (\Box_z \alpha \wedge \bigwedge_{j \leq m_z} \Diamond_z \beta_j) \bigwedge_{z \in \{x,y\}} (\forall z\, \gamma \wedge \bigwedge_{k \leq n_z} \exists z\, \delta_k) \wedge \forall x \forall y\, \varphi \wedge \bigwedge_{l \leq b} \forall x \exists y\, \psi_l$$

Bounded model construction for PTML^2

Suppose $\mathcal{M}, r \models \theta$ where θ is in the Fine Scott normal form.

$$\bigwedge_{i \leq a} s_i \wedge \bigwedge_{z \in \{x,y\}} (\Box_z \alpha \wedge \bigwedge_{j \leq m_z} \Diamond_z \beta_j) \bigwedge_{z \in \{x,y\}} (\forall z\, \gamma \wedge \bigwedge_{k \leq n_z} \exists z\, \delta_k) \wedge \forall x \forall y\, \varphi \wedge \bigwedge_{l \leq b} \forall x \exists y\, \psi_l$$

For every $w \in \mathcal{W}$

- Let $\text{1-type}(w) = \{\text{1-type}(w, c) \mid c \in \delta(w)\}$

Bounded model construction for PTML^2

Suppose $\mathcal{M}, r \models \theta$ where θ is in the Fine Scott normal form.

$$\bigwedge_{i\leq a} s_i \wedge \bigwedge_{z\in\{x,y\}} (\Box_z\alpha \wedge \bigwedge_{j\leq m_z} \Diamond_z\beta_j) \bigwedge_{z\in\{x,y\}} (\forall z\, \gamma \wedge \bigwedge_{k\leq n_z} \exists z\, \delta_k) \wedge \forall x \forall y\, \varphi \wedge \bigwedge_{l\leq b} \forall x \exists y\, \psi_l$$

For every $w \in \mathcal{W}$

- Let 1-type$(w) = \{$1-type$(w, c) \mid c \in \delta(w)\}$
- For every $\Lambda \in$ 1-type(w) let $a^w_\Lambda \in \delta(w)$ such that 1-type$(a^w_\Lambda) = \Lambda$.

Bounded model construction for PTML^2

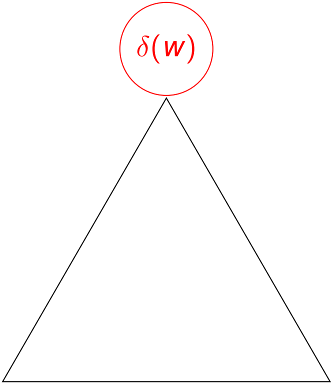
Suppose $\mathcal{M}, r \models \theta$ where θ is in the Fine Scott normal form.

$$\bigwedge_{i \leq a} s_i \wedge \bigwedge_{z \in \{x,y\}} (\Box_z \alpha \wedge \bigwedge_{j \leq m_z} \Diamond_z \beta_j) \bigwedge_{z \in \{x,y\}} (\forall z\, \gamma \wedge \bigwedge_{k \leq n_z} \exists z\, \delta_k) \wedge \forall x \forall y\, \varphi \wedge \bigwedge_{l \leq b} \forall x \exists y\, \psi_l$$

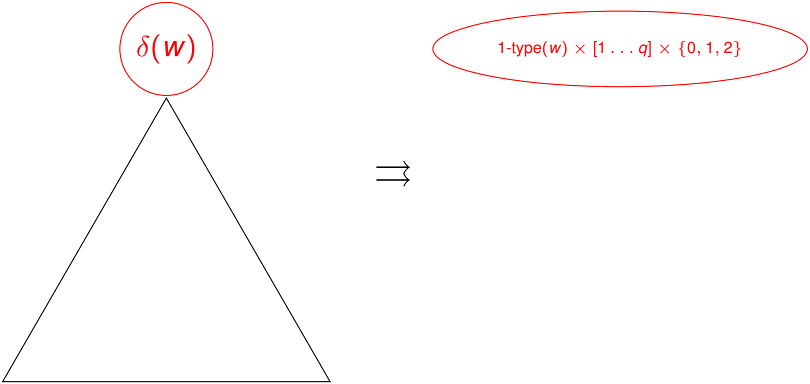
For every $w \in \mathcal{W}$

- Let $\text{1-type}(w) = \{\text{1-type}(w, c) \mid c \in \delta(w)\}$
- For every $\Lambda \in \text{1-type}(w)$ let $a^w_\Lambda \in \delta(w)$ such that $\text{1-type}(a^w_\Lambda) = \Lambda$.
- Let $E_\theta = \{\chi_1, \ldots \chi_q\}$ be the set of all subformulas of θ that need witnesses.

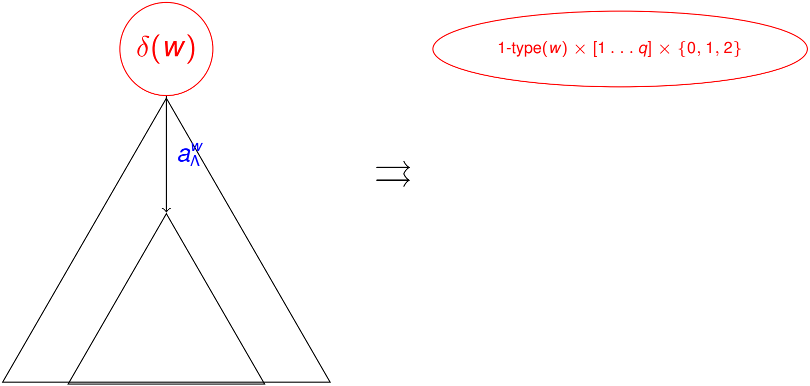
Bounded model construction for $PTML^2$



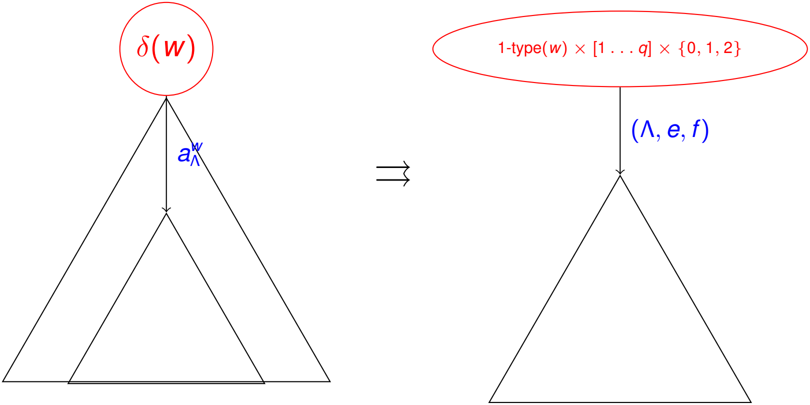
Bounded model construction for $PTML^2$



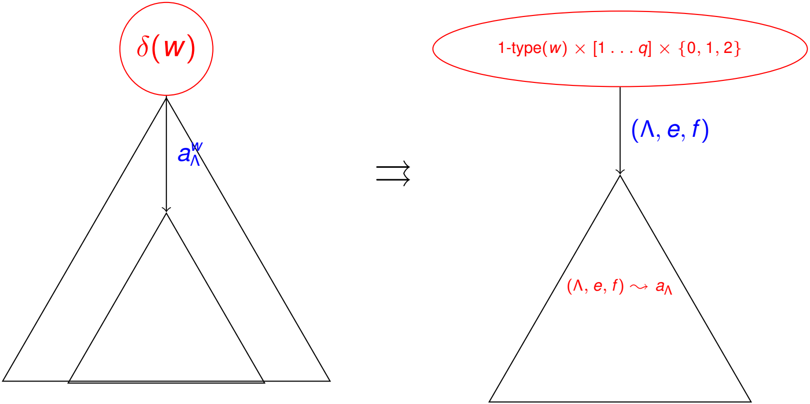
Bounded model construction for PTML^2



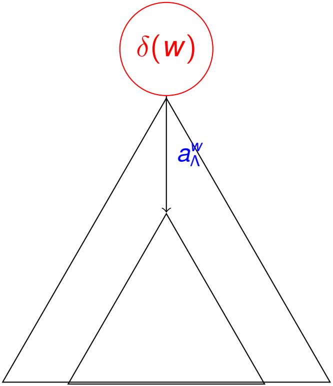
Bounded model construction for PTML^2



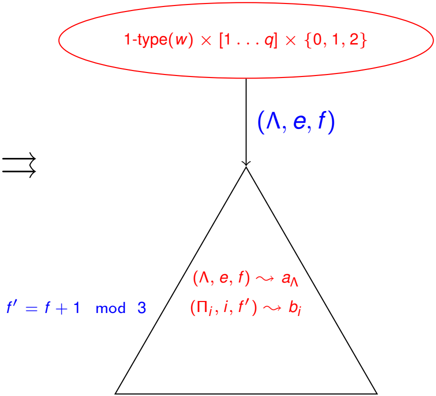
Bounded model construction for PTML^2



Bounded model construction for PTML^2

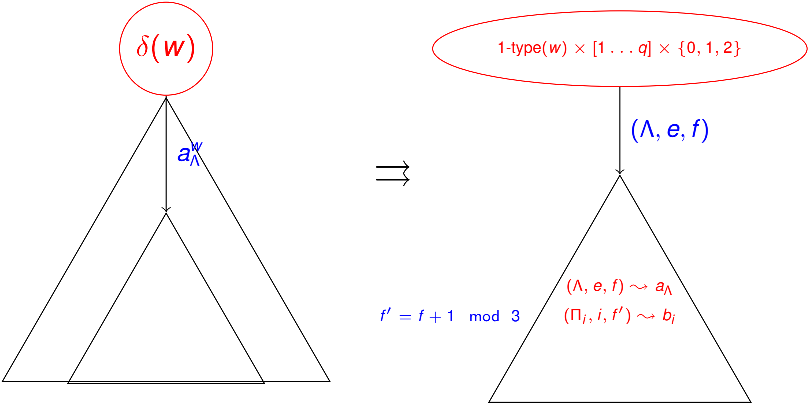


$\Rightarrow$



Let witnesses of $a_\wedge$ be $b_1, \ldots b_q$ and $1\text{-type}(w, b_i) = \Pi_i$

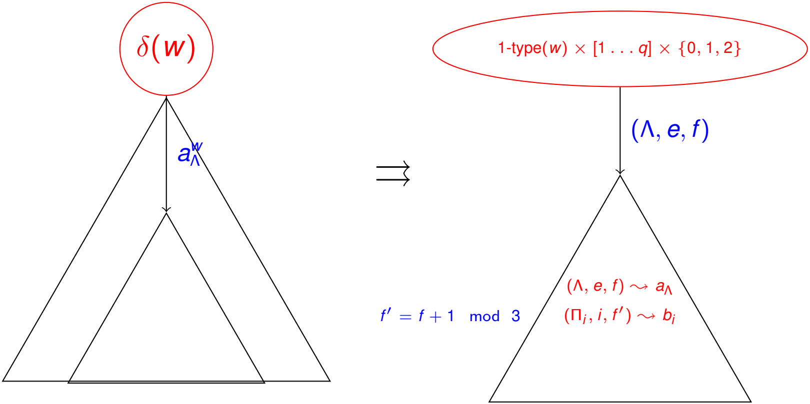
Bounded model construction for PTML^2



Let witnesses of a_Λ be $b_1, \ldots b_q$ and 1-type$(w, b_i) = \Pi_i$

For some a_Σ^w with wit. $d_1, \ldots d_q$ where 1-type$(w, d_l) = \Lambda$

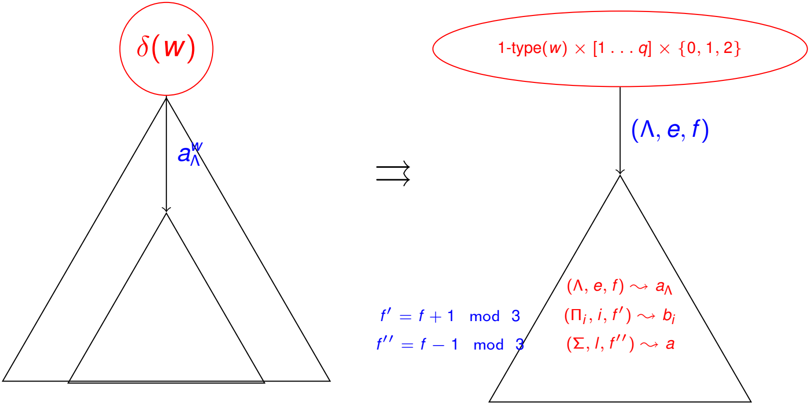
Bounded model construction for PTML^2



Let witnesses of a_Λ be $b_1, \ldots b_q$ and 1-type$(w, b_i) = \Pi_i$

For some a_Σ^w with wit. $d_1, \ldots d_q$ where 1-type$(w, d_l) = \Lambda$
There is some $a \in \delta(w)$ st 2-type$(w, d_l, a_\Sigma^w) =$ 2-type(w, a_Λ, a)

Bounded model construction for PTML^2



Let witnesses of a_Λ be $b_1, \ldots b_q$ and $1\text{-type}(w, b_i) = \Pi_i$

For some a_Σ^w with wit. $d_1, \ldots d_q$ where $1\text{-type}(w, d_l) = \Lambda$
There is some $a \in \delta(w)$ st $2\text{-type}(w, d_l, a_\Sigma^w) = 2\text{-type}(w, a_\Lambda, a)$

$PTML^2$ satisfiability

Theorem

Satisfiability problem for $PTML^2$ is in 2-EXPSPACE.

PTML^2 satisfiability

Theorem

Satisfiability problem for PTML^2 is in 2-EXPSPACE.

Proof.

- Every satisfiable formula can be satisfied in a model with $2^{2^{O(|\theta|)}}$ many agents.

$PTML^2$ satisfiability

Theorem

Satisfiability problem for $PTML^2$ is in 2-EXPSPACE.

Proof.

- Every satisfiable formula can be satisfied in a model with $2^{2^{O(|\theta|)}}$ many agents.
- Algorithm can guess the number of agents and expand the quantifiers as conjunction and disjunctions.

$PTML^2$ satisfiability

Theorem

Satisfiability problem for $PTML^2$ is in 2-EXPSPACE.

Proof.

- Every satisfiable formula can be satisfied in a model with $2^{2^{O(|\theta|)}}$ many agents.
- Algorithm can guess the number of agents and expand the quantifiers as conjunction and disjunctions.
- Reduces to propositional multi-modal formula satisfiability.

□

Satisfiability: Conclusion

- 2 variable TML (with arbitrary arity predicates) is as hard as 2 variable PTML
  Idea: Translate $P(x, y)$ to $\Diamond_x \Diamond_y (p)$

Satisfiability: Conclusion

- 2 variable TML (with arbitrary arity predicates) is as hard as 2 variable PTML
  Idea: Translate $P(x, y)$ to $\Diamond_x \Diamond_y (p)$
- Adding 1 constant to 2 variable fragment of PTML:Undecidable
  (Reduction from 2 variable FOML)
  $\Box \psi(x, y) \rightsquigarrow \Box_c \psi(x, y)$

Satisfiability: Conclusion

- 2 variable TML (with arbitrary arity predicates) is as hard as 2 variable PTML
  Idea: Translate $P(x, y)$ to $\Diamond_x \Diamond_y (p)$
- Adding 1 constant to 2 variable fragment of PTML:Undecidable
  (Reduction from 2 variable FOML)
  $\Box\psi(x, y) \rightsquigarrow \Box_c \psi(x, y)$
- Adding Equality?

Satisfiability: Conclusion

- 2 variable TML (with arbitrary arity predicates) is as hard as 2 variable PTML
  Idea: Translate $P(x, y)$ to $\Diamond_x \Diamond_y (p)$
- Adding 1 constant to 2 variable fragment of PTML:Undecidable
  (Reduction from 2 variable FOML)
  $\Box\psi(x, y) \rightsquigarrow \Box_c \psi(x, y)$
- Adding Equality?
- Frame restrictions.

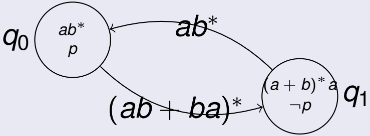
Satisfiability: Conclusion

- 2 variable TML (with arbitrary arity predicates) is as hard as 2 variable PTML
  Idea: Translate $P(x, y)$ to $\Diamond_x \Diamond_y (p)$
- Adding 1 constant to 2 variable fragment of PTML:Undecidable
  (Reduction from 2 variable FOML)
  $\Box\psi(x, y) \rightsquigarrow \Box_c\psi(x, y)$
- Adding Equality?
- Frame restrictions.
- Lower bound: NEXPTIME
  Upper bound: 2-EXPSPACE.

Model checking

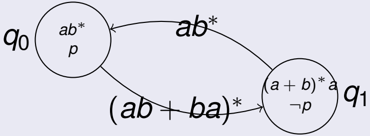
Model checking

Finite specification

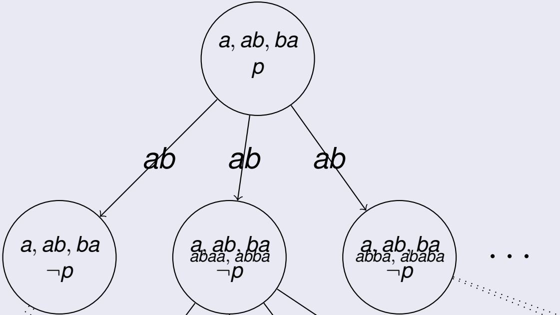


Model checking

Finite specification



Unravelling



THANK YOU