Automata Terms in a Lazy WS*k*S Decision Procedure

Vojtěch Havlena Lukáš Holík
Ondřej Lengál Tomáš Vojnar

Brno University of Technology
Czech Republic

30 August 2019 (CADE'19)

The WSkS Logic

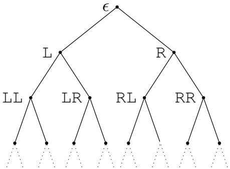
- Monadic Second-order $\rightsquigarrow$ quantification over sets

The WSkS Logic

- Monadic Second-order $\rightsquigarrow$ quantification over sets
- Weak $\rightsquigarrow$ finite models

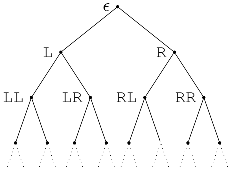
The WSkS Logic

- Monadic Second-order $\rightsquigarrow$ quantification over sets
- Weak $\rightsquigarrow$ finite models
- k successors $\rightsquigarrow$ assignments represent positions in an infinite k-ary tree



The WSkS Logic

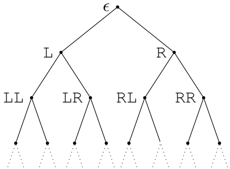
- Monadic Second-order $\rightsquigarrow$ quantification over sets
- Weak $\rightsquigarrow$ finite models
- k successors $\rightsquigarrow$ assignments represent positions in an infinite k-ary tree



- Application: e.g., reasoning about heap structures (STRAND), dec. proc. for separation logic, . . .
- Tool MONA [Klarlund et al.'98]

The WSkS Logic

- Monadic Second-order $\rightsquigarrow$ quantification over sets
- Weak $\rightsquigarrow$ finite models
- k successors $\rightsquigarrow$ assignments represent positions in an infinite k-ary tree



- Application: e.g., reasoning about heap structures (STRAND), dec. proc. for separation logic, . . .
- Tool MONA [Klarlund et al.'98]

- Closely related to finite tree automata [Doner'65]
- Decidable but **NONELEMENTARY**
  - Blow-up caused by quantifier alternations ($(\exists\forall)^*$)

Syntax and Semantics of WS2S

Syntax (restricted to $k = 2$)

$$\text{atom } \psi ::= X \subseteq Y \mid X = S_{\mathrm{L}}(Y) \mid X = S_{\mathrm{R}}(Y)$$
$$\text{formula } \varphi ::= \exists X.\, \varphi \mid \varphi \wedge \varphi \mid \neg\varphi \mid \psi$$

X, Y are second-order variables

Syntax and Semantics of WS2S

Syntax (restricted to $k = 2$)

$$\text{atom } \psi ::= X \subseteq Y \mid X = S_{\mathtt{L}}(Y) \mid X = S_{\mathtt{R}}(Y)$$
$$\text{formula } \varphi ::= \exists X.\, \varphi \mid \varphi \land \varphi \mid \neg\varphi \mid \psi$$

X, Y are second-order variables

Semantics

- Model of $\varphi(\mathbb{X})$ is an assignment $\eta : \mathbb{X} \to \mathcal{P}_\omega(\{\mathtt{L}, \mathtt{R}\}^*)$ satisfying φ
  - $\mathcal{P}_\omega(S) \rightsquigarrow$ set of all finite subsets of S
  - Example: $\eta = \{X \mapsto \{\mathtt{LRL}, \mathtt{RL}, \mathtt{RRL}\}\}$
- Assignment of a variable defines set of positions in a tree

Semantics of WS2S *cont.*

- Satisfaction of atoms under η

$$\eta \models X \subseteq Y \quad \text{iff} \quad \eta(X) \subseteq \eta(Y)$$
$$\eta \models X = S_{\mathtt{L}}(Y) \quad \text{iff} \quad \eta(X) = \{p.\mathtt{L} \mid p \in \eta(Y)\}$$
$$\eta \models X = S_{\mathtt{R}}(Y) \quad \text{iff} \quad \eta(X) = \{p.\mathtt{R} \mid p \in \eta(Y)\}$$

Example

(a) $\left\{ \begin{array}{l} X \mapsto \{\mathtt{LR}, \mathtt{R}, \mathtt{RR}\}, \\ Y \mapsto \{\mathtt{LRL}, \mathtt{RL}, \mathtt{RRL}\} \end{array} \right\} \models Y = S_{\mathtt{L}}(X)$ (b) $\left\{ \begin{array}{l} X \mapsto \{\mathtt{LR}, \mathtt{R}, \mathtt{RR}\}, \\ Y \mapsto \{\mathtt{LRR}, \mathtt{RR}, \mathtt{RRR}\} \end{array} \right\} \models Y = S_{\mathtt{R}}(X)$

Semantics of WS2S *cont.*

- Satisfaction of formulae under η

$$\eta \models \varphi_1 \land \varphi_2 \quad \text{iff} \quad \eta \models \varphi_1 \text{ and } \eta \models \varphi_2$$
$$\eta \models \neg\varphi \quad \text{iff} \quad \eta \not\models \varphi$$
$$\eta \models \exists X.\, \psi \quad \text{iff} \quad \text{exists } S \in \mathcal{P}_\omega(\{\mathtt{L}, \mathtt{R}\}^*) \text{ s.t. } \eta \cup \{X \mapsto S\} \models \psi$$

- Validity, satisfiability

Representing Models as Trees

- Model of $\varphi(X, Y)$ encoded as a finite tree of symbols $\boxed{X|Y}$
- Formula language $\mathcal{L}(\varphi) = \{\tau \mid \tau$ is an encoding of η s.t. $\eta \models \varphi\}$

Representing Models as Trees

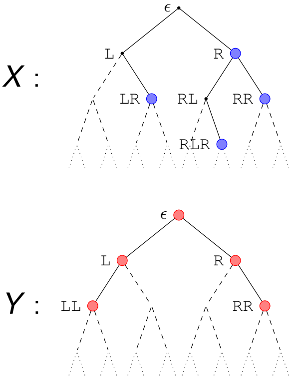
- Model of $\varphi(X, Y)$ encoded as a finite tree of symbols X|Y
- Formula language $\mathcal{L}(\varphi) = \{\tau \mid \tau$ is an encoding of η s.t. $\eta \models \varphi\}$

- Example: $\eta = \{X \mapsto \{\mathtt{LR}, \mathtt{R}, \mathtt{RLR}, \mathtt{RR}\}, Y \mapsto \{\epsilon, \mathtt{L}, \mathtt{LL}, \mathtt{R}, \mathtt{RR}\}\}$

Representing Models as Trees

- Model of $\varphi(X, Y)$ encoded as a finite tree of symbols $\boxed{X|Y}$
- Formula language $\mathcal{L}(\varphi) = \{\tau \mid \tau$ is an encoding of η s.t. $\eta \models \varphi\}$

- Example: $\eta = \{X \mapsto \{\mathtt{LR}, \mathtt{R}, \mathtt{RLR}, \mathtt{RR}\}, Y \mapsto \{\epsilon, \mathtt{L}, \mathtt{LL}, \mathtt{R}, \mathtt{RR}\}\}$

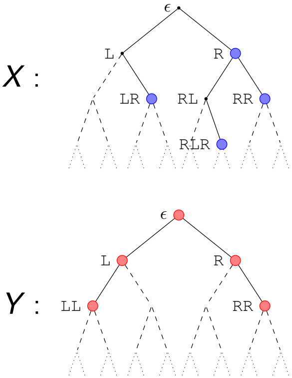


(a) Positions assigned to X,Y

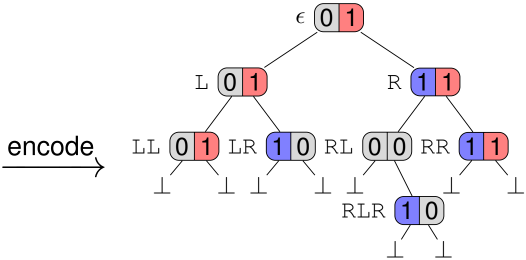
Representing Models as Trees

- Model of $\varphi(X, Y)$ encoded as a finite tree of symbols $\begin{bmatrix} X \mid Y \end{bmatrix}$
- Formula language $\mathcal{L}(\varphi) = \{\tau \mid \tau \text{ is an encoding of } \eta \text{ s.t. } \eta \models \varphi\}$

- Example: $\eta = \{X \mapsto \{\mathtt{LR}, \mathtt{R}, \mathtt{RLR}, \mathtt{RR}\}, Y \mapsto \{\epsilon, \mathtt{L}, \mathtt{LL}, \mathtt{R}, \mathtt{RR}\}\}$



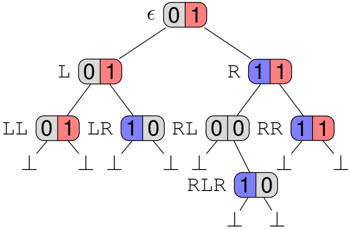
(a) Positions assigned to X,Y



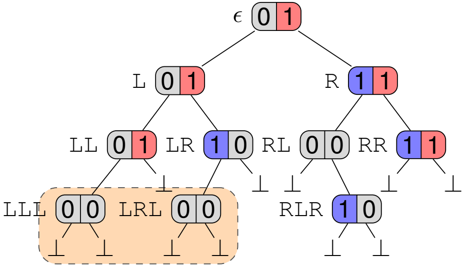
(b) Encoding of η into a binary tree

Representing Models as Trees

- Example: $\eta = \{X \mapsto \{\mathtt{LR}, \mathtt{R}, \mathtt{RLR}, \mathtt{RR}\}, Y \mapsto \{\epsilon, \mathtt{L}, \mathtt{LL}, \mathtt{R}, \mathtt{RR}\}\}$



(a) Minimal encoding



(b) Nonminimal encoding

- More encodings:
  - minimal encoding with no $\vec{0}$-labelled subtrees ($\vec{0}$ = 0 . . . 0)
  - encodings obtained from minimal by appending $\vec{0}$-labelled subtrees

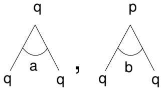
Tree Automaton

- Concise representation of a set of trees
- Finite Tree Automaton
  - Finite set of states Q
  - Set of leaf states $I \subseteq Q$
  - Set of root states $R \subseteq Q$
  - Transition function $\Delta_a : Q \times Q \to Q$ for each symbol a.

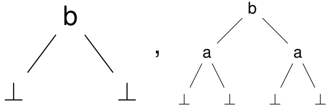
Tree Automaton

- Concise representation of a set of trees
- Finite Tree Automaton
  - Finite set of states Q
  - Set of leaf states $I \subseteq Q$
  - Set of root states $R \subseteq Q$
  - Transition function $\Delta_a : Q \times Q \to Q$ for each symbol a.

- Example: $\mathcal{A} = \Big(\overbrace{\{p, q\}}^{\text{states}}, \underbrace{\{q\}}_{\text{leaf states}}, \overbrace{\{p\}}^{\text{root states}}, \underbrace{\{ \ldots \}}_{\text{transitions}}\Big)$



  - $\mathcal{L}(\mathcal{A}) = \{ \ldots, \ldots \}$



Decision Procedure

- For a formula φ inductively construct TA $\mathcal{A}_\varphi$ s.t. $\mathcal{L}(\varphi) = \mathcal{L}(\mathcal{A}_\varphi)$
  - For φ being an atom $\rightsquigarrow$ use predefined TA $\mathcal{A}_\varphi$
  - For $\varphi = \psi_1 \land \psi_2 \rightsquigarrow \mathcal{A}_\varphi = \mathcal{A}_{\psi_1} \cap \mathcal{A}_{\psi_2}$
  - For $\varphi = \neg\psi \rightsquigarrow \mathcal{A}_\varphi = \mathcal{A}_\psi^\complement$ (requires determinization)
  - For $\varphi = \exists X.\, \psi \rightsquigarrow \mathcal{A}_\varphi = \pi_X(\mathcal{A}_\psi) - \vec{0}^*$ (projection, saturation)

Decision Procedure

- For a formula φ inductively construct TA $\mathcal{A}_\varphi$ s.t. $\mathcal{L}(\varphi) = \mathcal{L}(\mathcal{A}_\varphi)$
  - For φ being an atom $\rightsquigarrow$ use predefined TA $\mathcal{A}_\varphi$
  - For $\varphi = \psi_1 \wedge \psi_2$ $\rightsquigarrow$ $\mathcal{A}_\varphi = \mathcal{A}_{\psi_1} \cap \mathcal{A}_{\psi_2}$
  - For $\varphi = \neg\psi$ $\rightsquigarrow$ $\mathcal{A}_\varphi = \mathcal{A}_\psi^{\complement}$ (requires determinization)
  - For $\varphi = \exists X.\, \psi$ $\rightsquigarrow$ $\mathcal{A}_\varphi = \pi_X(\mathcal{A}_\psi) - \vec{0}^{*}$ (projection, saturation)
- Satisfiability checking $\mathcal{L}(\mathcal{A}_\varphi) \stackrel{?}{=} \emptyset$

Decision Procedure

- For a formula φ inductively construct TA $\mathcal{A}_\varphi$ s.t. $\mathcal{L}(\varphi) = \mathcal{L}(\mathcal{A}_\varphi)$
  - For φ being an atom $\rightsquigarrow$ use predefined TA $\mathcal{A}_\varphi$
  - For $\varphi = \psi_1 \wedge \psi_2$ $\rightsquigarrow$ $\mathcal{A}_\varphi = \mathcal{A}_{\psi_1} \cap \mathcal{A}_{\psi_2}$
  - For $\varphi = \neg\psi$ $\rightsquigarrow$ $\mathcal{A}_\varphi = \mathcal{A}_\psi^{\complement}$ (requires determinization)
  - For $\varphi = \exists X.\, \psi$ $\rightsquigarrow$ $\mathcal{A}_\varphi = \pi_X(\mathcal{A}_\psi) - \vec{0}^{*}$ (projection, saturation)
- Satisfiability checking $\mathcal{L}(\mathcal{A}_\varphi) \stackrel{?}{=} \emptyset$

Projection π_X

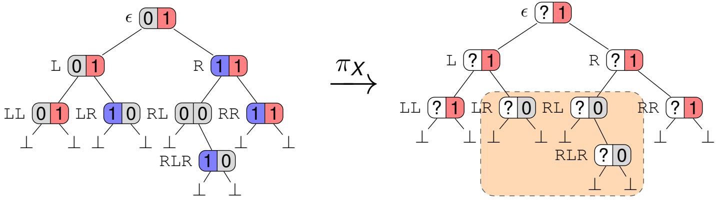
- Ignore X-track of each symbol

Decision Procedure

- For a formula φ inductively construct TA $\mathcal{A}_\varphi$ s.t. $\mathcal{L}(\varphi) = \mathcal{L}(\mathcal{A}_\varphi)$
  - For φ being an atom $\rightsquigarrow$ use predefined TA $\mathcal{A}_\varphi$
  - For $\varphi = \psi_1 \wedge \psi_2$ $\rightsquigarrow$ $\mathcal{A}_\varphi = \mathcal{A}_{\psi_1} \cap \mathcal{A}_{\psi_2}$
  - For $\varphi = \neg\psi$ $\rightsquigarrow$ $\mathcal{A}_\varphi = \mathcal{A}_\psi^{\complement}$ (requires determinization)
  - For $\varphi = \exists X.\, \psi$ $\rightsquigarrow$ $\mathcal{A}_\varphi = \pi_X(\mathcal{A}_\psi) - \vec{0}^*$ (projection, saturation)
- Satisfiability checking $\mathcal{L}(\mathcal{A}_\varphi) \stackrel{?}{=} \emptyset$

Projection π_X

- Ignore X-track of each symbol

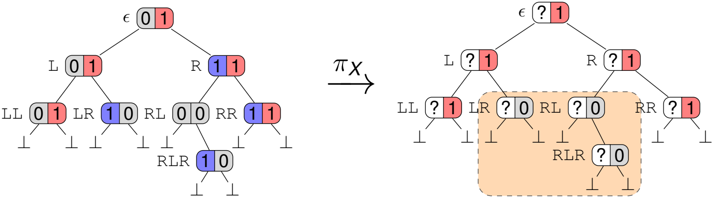


Decision Procedure

- For a formula φ inductively construct TA $\mathcal{A}_\varphi$ s.t. $\mathcal{L}(\varphi) = \mathcal{L}(\mathcal{A}_\varphi)$
  - For φ being an atom $\rightsquigarrow$ use predefined TA $\mathcal{A}_\varphi$
  - For $\varphi = \psi_1 \wedge \psi_2$ $\rightsquigarrow$ $\mathcal{A}_\varphi = \mathcal{A}_{\psi_1} \cap \mathcal{A}_{\psi_2}$
  - For $\varphi = \neg\psi$ $\rightsquigarrow$ $\mathcal{A}_\varphi = \mathcal{A}_\psi^{\complement}$ (requires determinization)
  - For $\varphi = \exists X.\, \psi$ $\rightsquigarrow$ $\mathcal{A}_\varphi = \pi_X(\mathcal{A}_\psi) - \vec{0}^*$ (projection, saturation)
- Satisfiability checking $\mathcal{L}(\mathcal{A}_\varphi) \stackrel{?}{=} \emptyset$

Projection π_X

- Ignore X-track of each symbol

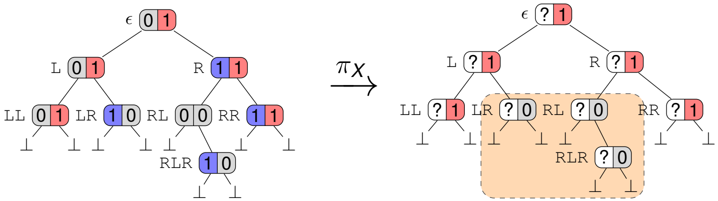


- For TA on the level of transition function

Decision Procedure *cont.*

Projection π_X

- Ignore X-track of each symbol

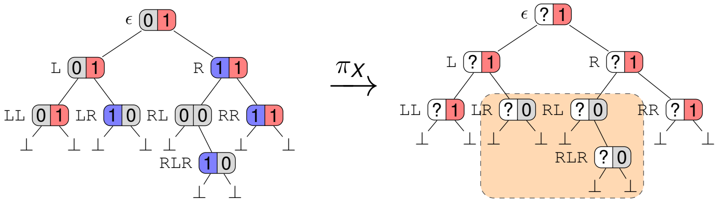


- For TA on the level of transition function

Decision Procedure *cont.*

Projection π_X

- Ignore X-track of each symbol



- For TA on the level of transition function

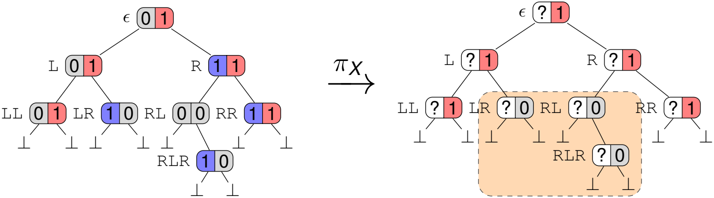
Saturation – $\vec{0}^*$

- Projection can prevent from accepting all encodings

Decision Procedure *cont.*

Projection π_X

- Ignore X-track of each symbol



- For TA on the level of transition function

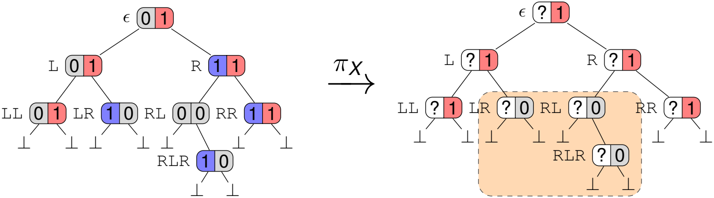
Saturation $-\vec{0}^{*}$

- Projection can prevent from accepting all encodings
- Remove zero-labelled subtrees

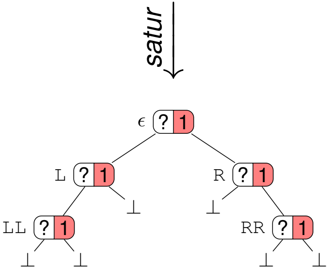
Decision Procedure *cont.*

Projection π_X

- Ignore X-track of each symbol



- For TA on the level of transition function



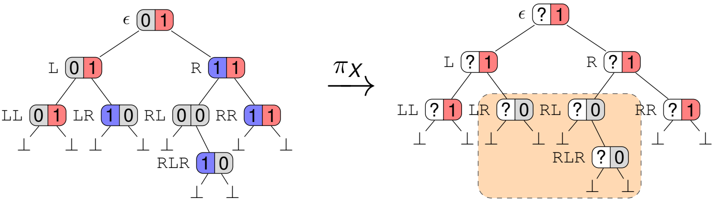
Saturation $- \vec{0}^*$

- Projection can prevent from accepting all encodings
- Remove zero-labelled subtrees

Decision Procedure *cont.*

Projection π_X

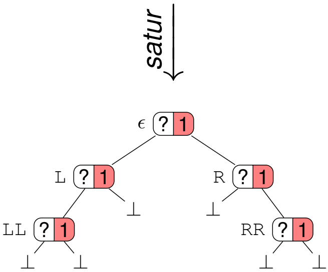
- Ignore X-track of each symbol



- For TA on the level of transition function

Saturation $- \vec{0}^*$

- Projection can prevent from accepting all encodings
- Remove zero-labelled subtrees
- For TA saturate the set of leaf states $\rightsquigarrow$ reachability from leaf states over $\vec{0}$



Towards Automata Terms

- Satisfiability checking of φ
  $\rightsquigarrow$ construct the whole TA $\mathcal{A}_\varphi$
  $\rightsquigarrow$ check $\mathcal{L}(\mathcal{A}_\varphi) = \emptyset$

Towards Automata Terms

- Satisfiability checking of φ
  - $\rightsquigarrow$ construct the whole TA $\mathcal{A}_\varphi$
  - $\rightsquigarrow$ check $\mathcal{L}(\mathcal{A}_\varphi) = \emptyset$

$$\varphi \equiv \neg\exists X.\ \mathit{Sing}(X) \land X = \{\epsilon\}$$

Towards Automata Terms

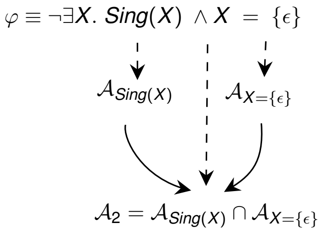
- Satisfiability checking of φ
  - $\rightsquigarrow$ construct the whole TA $\mathcal{A}_\varphi$
  - $\rightsquigarrow$ check $\mathcal{L}(\mathcal{A}_\varphi) = \emptyset$

$$\varphi \equiv \neg\exists X.\ \mathit{Sing}(X) \wedge X = \{\epsilon\}$$

$$\mathcal{A}_{\mathit{Sing}(X)} \qquad \mathcal{A}_{X=\{\epsilon\}}$$

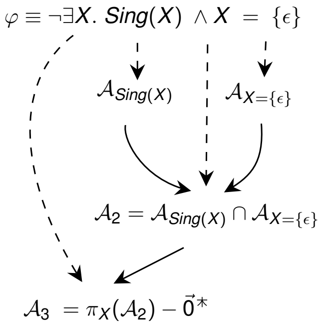
Towards Automata Terms

- Satisfiability checking of φ
  - $\rightsquigarrow$ construct the whole TA $\mathcal{A}_\varphi$
  - $\rightsquigarrow$ check $\mathcal{L}(\mathcal{A}_\varphi) = \emptyset$



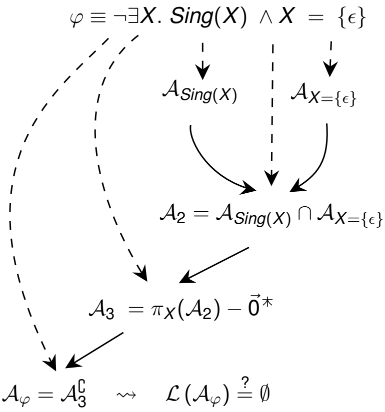
Towards Automata Terms

- Satisfiability checking of φ
  - $\rightsquigarrow$ construct the whole TA $\mathcal{A}_\varphi$
  - $\rightsquigarrow$ check $\mathcal{L}(\mathcal{A}_\varphi) = \emptyset$



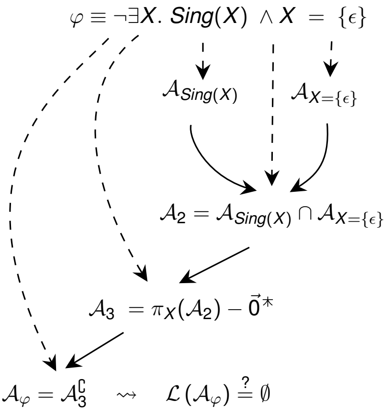
Towards Automata Terms

- Satisfiability checking of φ
  - $\rightsquigarrow$ construct the whole TA $\mathcal{A}_\varphi$
  - $\rightsquigarrow$ check $\mathcal{L}(\mathcal{A}_\varphi) = \emptyset$



Towards Automata Terms

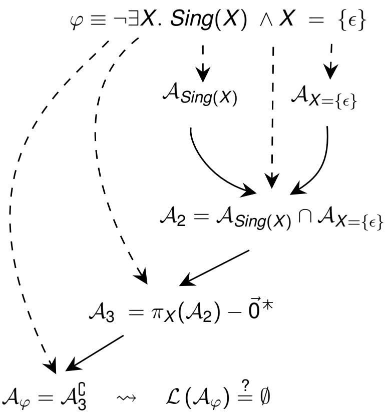
- Satisfiability checking of φ
  - $\rightsquigarrow$ construct the whole TA $\mathcal{A}_\varphi$
  - $\rightsquigarrow$ check $\mathcal{L}(\mathcal{A}_\varphi) = \emptyset$



- Quantifier alternation can cause exponential blow-up

Towards Automata Terms

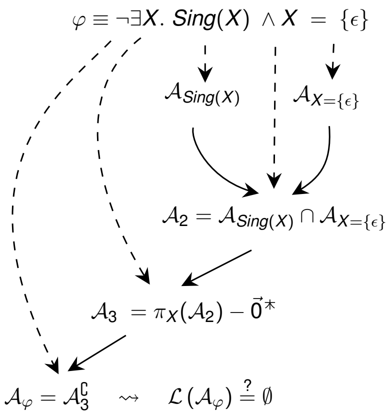
- Satisfiability checking of φ
  $\rightsquigarrow$ construct the whole TA $\mathcal{A}_\varphi$
  $\rightsquigarrow$ check $\mathcal{L}(\mathcal{A}_\varphi) = \emptyset$



- Quantifier alternation can cause exponential blow-up
- What if for $\psi_1 \wedge \psi_2$ we have $\mathcal{L}(\mathcal{A}_{\psi_1}) = \emptyset$?

Towards Automata Terms

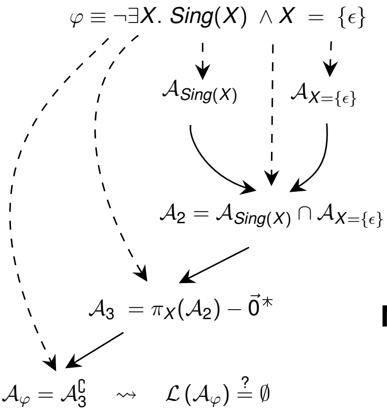
- Satisfiability checking of φ
  - $\rightsquigarrow$ construct the whole TA $\mathcal{A}_\varphi$
  - $\rightsquigarrow$ check $\mathcal{L}(\mathcal{A}_\varphi) = \emptyset$



- Quantifier alternation can cause exponential blow-up
- What if for $\psi_1 \land \psi_2$ we have $\mathcal{L}(\mathcal{A}_{\psi_1}) = \emptyset$?
  - No need to construct $\mathcal{A}_{\psi_2}$ and $\mathcal{A}_{\psi_1 \land \psi_2}$

Towards Automata Terms

- Satisfiability checking of φ
  $\rightsquigarrow$ construct the whole TA $\mathcal{A}_\varphi$
  $\rightsquigarrow$ check $\mathcal{L}(\mathcal{A}_\varphi) = \emptyset$



- Quantifier alternation can cause exponential blow-up
- What if for $\psi_1 \wedge \psi_2$ we have $\mathcal{L}(\mathcal{A}_{\psi_1}) = \emptyset$?
  - No need to construct $\mathcal{A}_{\psi_2}$ and $\mathcal{A}_{\psi_1 \wedge \psi_2}$

No need to construct the whole TA $\rightsquigarrow$ construction directed by emptiness check

Automata Terms Overview

- Implicit representation of TAs constructed by automata operations
  - Tracking the information about used automata operations
  - Terms represent
    - states (e.g., t_1 & t_2)
    - a set of a TA leaf states (e.g., $\{t_1, t_2, t_3\}$, $\{t_1, t_2\} - \vec{0}^{\ast}$)
  - Term leaves are states of a base TA
  - Term transition function and term root predicate defined inductively on the structure of terms

Automata Terms Overview

- Implicit representation of TAs constructed by automata operations
  - Tracking the information about used automata operations
  - Terms represent
    - states (e.g., t_1 & t_2)
    - a set of a TA leaf states (e.g., $\{t_1, t_2, t_3\}$, $\{t_1, t_2\} - \vec{0}^{\ast}$)
  - Term leaves are states of a base TA
  - Term transition function and term root predicate defined inductively on the structure of terms

- Automata terms allow to construct parts of automata from incomplete parts on lower levels (in contrast to the classical proc.)
  - Even if the lower parts are not finished yet

Automata Terms Overview

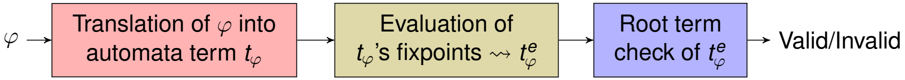
- Implicit representation of TAs constructed by automata operations
  - Tracking the information about used automata operations
  - Terms represent
    - states (e.g., t_1 & t_2)
    - a set of a TA leaf states (e.g., $\{t_1, t_2, t_3\}$, $\{t_1, t_2\} - \vec{0}^{\ast}$)
  - Term leaves are states of a base TA
  - Term transition function and term root predicate defined inductively on the structure of terms

- Automata terms allow to construct parts of automata from incomplete parts on lower levels (in contrast to the classical proc.)
  - Even if the lower parts are not finished yet

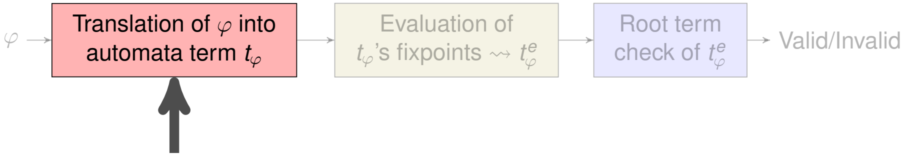
- Allows to prune the state space and to test emptiness on the fly $\rightsquigarrow$ Focus on ground formulae

$$\models \varphi \iff \perp \in \mathcal{L}(\varphi)$$

Decision Procedure



Overview



φ
Translation of φ into automata term t_φ
Evaluation of t_φ's fixpoints $\rightsquigarrow t^e_\varphi$
Root term check of t^e_φ
Valid/Invalid

Automata Terms

- Convert formula φ to automata term t_φ
  - φ is atom $\rightsquigarrow$ t_φ is the set of leaf states of a base $\mathcal{A}_\varphi$
  - $\varphi = \psi_1 \wedge \psi_2$ $\rightsquigarrow$ $t_\varphi = t_{\psi_1} \,\&\, t_{\psi_2}$
  - $\varphi = \neg\psi$ $\rightsquigarrow$ $t_\varphi = \overline{t_\psi}$
  - $\varphi = \exists X.\, \psi$ $\rightsquigarrow$ $t_\varphi = \{\pi_X(t_\psi)\} - \vec{0}^*$
    $\rightsquigarrow$ Symbolic representation of unfinished fixpoint computation (saturation)

Automata Terms

- Convert formula φ to automata term t_φ
  - φ is atom $\rightsquigarrow$ t_φ is the set of leaf states of a base $\mathcal{A}_\varphi$
  - $\varphi = \psi_1 \wedge \psi_2$ $\rightsquigarrow$ $t_\varphi = t_{\psi_1} \,\&\, t_{\psi_2}$
  - $\varphi = \neg\psi$ $\rightsquigarrow$ $t_\varphi = \overline{t_\psi}$
  - $\varphi = \exists X.\, \psi$ $\rightsquigarrow$ $t_\varphi = \{\pi_X(t_\psi)\} - \vec{0}^*$
    $\rightsquigarrow$ Symbolic representation of unfinished fixpoint computation (saturation)

Example

- Formula $\varphi \equiv \neg\exists X.\, Sing(X) \wedge X = \{\epsilon\}$

Automata Terms

- Convert formula φ to automata term t_φ
  - φ is atom $\rightsquigarrow$ t_φ is the set of leaf states of a base $\mathcal{A}_\varphi$
  - $\varphi = \psi_1 \wedge \psi_2$ $\rightsquigarrow$ $t_\varphi = t_{\psi_1} \,\&\, t_{\psi_2}$
  - $\varphi = \neg\psi$ $\rightsquigarrow$ $t_\varphi = \overline{t_\psi}$
  - $\varphi = \exists X.\, \psi$ $\rightsquigarrow$ $t_\varphi = \{\pi_X(t_\psi)\} - \vec{0}^{\ast}$
    $\rightsquigarrow$ Symbolic representation of unfinished fixpoint computation (saturation)

Example

- Formula $\varphi \equiv \neg\exists X.\, Sing(X) \wedge X = \{\epsilon\}$
- Corresponding automata term $t_\varphi = \left\{ \overline{\{\pi_X(\{q_0\} \,\&\, \{p_0\})\} - \vec{0}^{\ast}} \right\}$

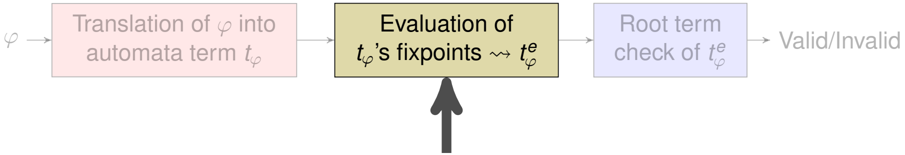
Automata Terms

- Convert formula φ to automata term t_φ
  - φ is atom $\rightsquigarrow$ t_φ is the set of leaf states of a base $\mathcal{A}_\varphi$
  - $\varphi = \psi_1 \wedge \psi_2$ $\rightsquigarrow$ $t_\varphi = t_{\psi_1} \,\&\, t_{\psi_2}$
  - $\varphi = \neg\psi$ $\rightsquigarrow$ $t_\varphi = \overline{t_\psi}$
  - $\varphi = \exists X.\, \psi$ $\rightsquigarrow$ $t_\varphi = \{\pi_X(t_\psi)\} - \vec{0}^*$
    $\rightsquigarrow$ Symbolic representation of unfinished fixpoint computation (saturation)

Example

- Formula $\varphi \equiv \neg\exists X.\, Sing(X) \wedge X = \{\epsilon\}$
- Corresponding automata term $t_\varphi = \left\{ \overline{\{\pi_X(\{q_0\} \,\&\, \{p_0\})\} - \vec{0}^*} \right\}$
- t_φ denotes the TA $\left(\pi_X(\mathcal{A}_{Sing(X)} \cap \mathcal{A}_{X=\{\epsilon\}}) - \vec{0}^*\right)^{\complement}$

Overview


φ
Translation of φ into automata term t_φ
Evaluation of t_φ's fixpoints $\rightsquigarrow t^e_\varphi$
Root term check of t^e_φ
Valid/Invalid

Fixpoint Evaluation

- The term $\underbrace{\{\pi_X(t_\psi)\}}_{S} - \vec{0}^*$ of formula $\exists X.\ \psi$ symbolically represents unfinished fixpoint computation
- Corresponds to the TA saturation $\pi_X(\mathcal{A}_\psi) - \vec{0}^*$

Fixpoint Evaluation

- The term $\underbrace{\{\pi_X(t_\psi)\}}_{S} - \vec{0}^{*}$ of formula $\exists X.\ \psi$ symbolically represents unfinished fixpoint computation
- Corresponds to the TA saturation $\pi_X(\mathcal{A}_\psi) - \vec{0}^{*}$
- The term $S - \vec{0}^{*}$ represents all terms bottom-up reachable from S via transition function Δ over symbol $\vec{0}$
  - Δ_a is a binary tree term transition function over a
  - $S - \vec{0}^{*} = \mu Z.\ S \cup \bigcup_{q,r \in Z} \Delta_{\vec{0}}(q, r)$

Fixpoint Evaluation

- The term $\underbrace{\{\pi_X(t_\psi)\}}_{S} - \vec{0}^*$ of formula $\exists X.\ \psi$ symbolically represents unfinished fixpoint computation
- Corresponds to the TA saturation $\pi_X(\mathcal{A}_\psi) - \vec{0}^*$
- The term $S - \vec{0}^*$ represents all terms bottom-up reachable from S via transition function Δ over symbol $\vec{0}$
  - Δ_a is a binary tree term transition function over a
  - $S - \vec{0}^* = \mu Z.\ S \cup \bigcup_{q,r \in Z} \Delta_{\vec{0}}(q, r)$

Example

1. $\{\, t \,\} - \vec{0}^* =$

Fixpoint Evaluation

- The term $\underbrace{\{\pi_X(t_\psi)\}}_{S} - \vec{0}^*$ of formula $\exists X.\ \psi$ symbolically represents unfinished fixpoint computation
- Corresponds to the TA saturation $\pi_X(\mathcal{A}_\psi) - \vec{0}^*$
- The term $S - \vec{0}^*$ represents all terms bottom-up reachable from S via transition function Δ over symbol $\vec{0}$
  - Δ_a is a binary tree term transition function over a
  - $S - \vec{0}^* = \mu Z.\ S \cup \bigcup_{q,r \in Z} \Delta_{\vec{0}}(q, r)$

Example

1. $\{ t \} - \vec{0}^* =$
   - $\Delta_{\vec{0}}(t , t) = \{ s \}$

Fixpoint Evaluation

- The term $\underbrace{\{\pi_X(t_\psi)\}}_{S} - \vec{0}^*$ of formula $\exists X.\ \psi$ symbolically represents unfinished fixpoint computation
- Corresponds to the TA saturation $\pi_X(\mathcal{A}_\psi) - \vec{0}^*$
- The term $S - \vec{0}^*$ represents all terms bottom-up reachable from S via transition function Δ over symbol $\vec{0}$
  - Δ_a is a binary tree term transition function over a
  - $S - \vec{0}^* = \mu Z.\ S \cup \bigcup_{q,r \in Z} \Delta_{\vec{0}}(q, r)$

Example

1. $\{t\} - \vec{0}^* = \{t, s\} - \vec{0}^*$
   - $\Delta_{\vec{0}}(t, t) = \{s\}$

Fixpoint Evaluation

- The term $\underbrace{\{\pi_X(t_\psi)\}}_{S} - \vec{0}^*$ of formula $\exists X.\ \psi$ symbolically represents unfinished fixpoint computation
- Corresponds to the TA saturation $\pi_X(\mathcal{A}_\psi) - \vec{0}^*$
- The term $S - \vec{0}^*$ represents all terms bottom-up reachable from S via transition function Δ over symbol $\vec{0}$
  - Δ_a is a binary tree term transition function over a
  - $S - \vec{0}^* = \mu Z.\ S \cup \bigcup_{q,r \in Z} \Delta_{\vec{0}}(q, r)$

Example

1. $\{t\} - \vec{0}^* = \{t, s\} - \vec{0}^*$
   - $\Delta_{\vec{0}}(t, t) = \{s\}$
2. $\{t, s\} - \vec{0}^* =$

Fixpoint Evaluation

- The term $\underbrace{\{\pi_X(t_\psi)\}}_{S} - \vec{0}^*$ of formula $\exists X.\ \psi$ symbolically represents unfinished fixpoint computation
- Corresponds to the TA saturation $\pi_X(\mathcal{A}_\psi) - \vec{0}^*$
- The term $S - \vec{0}^*$ represents all terms bottom-up reachable from S via transition function Δ over symbol $\vec{0}$
  - Δ_a is a binary tree term transition function over a
  - $S - \vec{0}^* = \mu Z.\ S \cup \bigcup_{q,r \in Z} \Delta_{\vec{0}}(q, r)$

Example

1. $\{t\} - \vec{0}^* = \{t, s\} - \vec{0}^*$
   - $\Delta_{\vec{0}}(t, t) = \{s\}$
2. $\{t, s\} - \vec{0}^* =$
   - $\Delta_{\vec{0}}(t, s) = \{t, s\}$

Fixpoint Evaluation

- The term $\underbrace{\{\pi_X(t_\psi)\}}_{S} - \vec{0}^*$ of formula $\exists X.\ \psi$ symbolically represents unfinished fixpoint computation
- Corresponds to the TA saturation $\pi_X(\mathcal{A}_\psi) - \vec{0}^*$
- The term $S - \vec{0}^*$ represents all terms bottom-up reachable from S via transition function Δ over symbol $\vec{0}$
  - Δ_a is a binary tree term transition function over a
  - $S - \vec{0}^* = \mu Z.\ S \cup \bigcup_{q,r \in Z} \Delta_{\vec{0}}(q, r)$

Example

1. $\{t\} - \vec{0}^* = \{t, s\} - \vec{0}^*$
   - $\Delta_{\vec{0}}(t, t) = \{s\}$
2. $\{t, s\} - \vec{0}^* =$
   - $\Delta_{\vec{0}}(t, s) = \{t, s\}$
   - $\Delta_{\vec{0}}(s, t) = \{t\}$

Fixpoint Evaluation

- The term $\underbrace{\{\pi_X(t_\psi)\}}_{S} - \vec{0}^*$ of formula $\exists X.\ \psi$ symbolically represents unfinished fixpoint computation
- Corresponds to the TA saturation $\pi_X(\mathcal{A}_\psi) - \vec{0}^*$
- The term $S - \vec{0}^*$ represents all terms bottom-up reachable from S via transition function Δ over symbol $\vec{0}$
  - Δ_a is a binary tree term transition function over a
  - $S - \vec{0}^* = \mu Z.\ S \cup \bigcup_{q,r \in Z} \Delta_{\vec{0}}(q, r)$

Example

1. $\{t\} - \vec{0}^* = \{t, s\} - \vec{0}^*$
   - $\Delta_{\vec{0}}(t, t) = \{s\}$
2. $\{t, s\} - \vec{0}^* =$
   - $\Delta_{\vec{0}}(t, s) = \{t, s\}$
   - $\Delta_{\vec{0}}(s, t) = \{t\}$
   - $\Delta_{\vec{0}}(s, s) = \{s\}$

Fixpoint Evaluation

- The term $\underbrace{\{\pi_X(t_\psi)\}}_{S} - \vec{0}^*$ of formula $\exists X.\ \psi$ symbolically represents unfinished fixpoint computation
- Corresponds to the TA saturation $\pi_X(\mathcal{A}_\psi) - \vec{0}^*$
- The term $S - \vec{0}^*$ represents all terms bottom-up reachable from S via transition function Δ over symbol $\vec{0}$
  - Δ_a is a binary tree term transition function over a
  - $S - \vec{0}^* = \mu Z.\ S \cup \bigcup_{q,r \in Z} \Delta_{\vec{0}}(q, r)$

Example

1. $\{t\} - \vec{0}^* = \{t, s\} - \vec{0}^*$
   - $\Delta_{\vec{0}}(t, t) = \{s\}$
2. $\{t, s\} - \vec{0}^* = \{t, s\} - \vec{0}^* = \{t, s\} \rightsquigarrow$ fixpoint reached
   - $\Delta_{\vec{0}}(t, s) = \{t, s\}$
   - $\Delta_{\vec{0}}(s, t) = \{t\}$
   - $\Delta_{\vec{0}}(s, s) = \{s\}$

Fixpoint Evaluation *ex.*

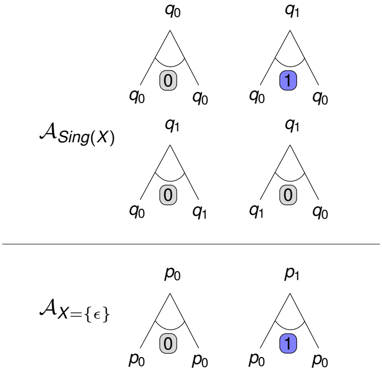
- Formula $\varphi \equiv \neg\exists X.\, Sing(X) \wedge X = \{\epsilon\}$ and its automata term
  $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{*}} \right\}$

Fixpoint Evaluation *ex.*

- Formula $\varphi \equiv \neg\exists X.\ \overline{Sing(X) \wedge X = \{\epsilon\}}$ and its automata term

$$t_\varphi = \left\{ \{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{\ast} \right\}$$

1. $\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{\ast} =$



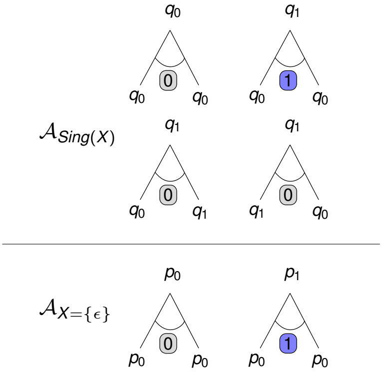
Fixpoint Evaluation *ex.*

- Formula $\varphi \equiv \neg\exists X.\ \overline{Sing(X) \wedge X = \{\epsilon\}}$ and its automata term

$$t_\varphi = \left\{ \{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* \right\}$$

1. $\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* =$
   - $\Delta_{\vec{0}}(\pi_X(q_0 \,\&\, p_0), \pi_X(q_0 \,\&\, p_0)) = ?$

   $\Delta_{\vec{0}}(\pi_X(q_0 \,\&\, p_0), \pi_X(q_0 \,\&\, p_0)) =$



Fixpoint Evaluation *ex.*

- Formula $\varphi \equiv \neg\exists X.\ \overline{Sing(X) \wedge X = \{\epsilon\}}$ and its automata term

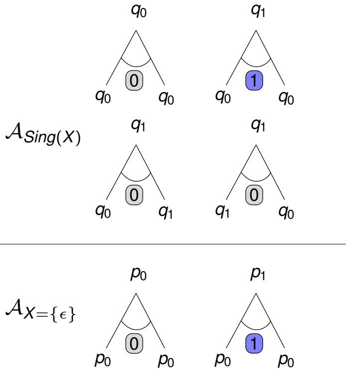
$$t_\varphi = \left\{ \boxed{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*} \right\}$$

1. $\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* =$
   - $\Delta_{\vec{0}}(\pi_X(q_0 \,\&\, p_0), \pi_X(q_0 \,\&\, p_0)) = ?$

$\Delta_{\vec{0}}(\pi_X(q_0 \,\&\, p_0), \pi_X(q_0 \,\&\, p_0)) =$

$\Delta_{\vec{0}}(\pi_X(t_1), \pi_X(t_2)) \rightsquigarrow \{\pi_X(\Delta_{X=0}(t_1, t_2)), \pi_X(\Delta_{X=1}(t_1, t_2))\}$

$\{\pi_X(\Delta_{X=1}(q_0 \,\&\, p_0, q_0 \,\&\, p_0)), \pi_X(\Delta_{X=0}(\dots))\} =$



Fixpoint Evaluation *ex.*

- Formula $\varphi \equiv \neg\exists X.\ \overline{Sing(X) \wedge X = \{\epsilon\}}$ and its automata term

$$t_\varphi = \left\{ \{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* \right\}$$

1. $\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* =$
   - $\Delta_{\vec{0}}(\pi_X(q_0 \,\&\, p_0), \pi_X(q_0 \,\&\, p_0)) = ?$

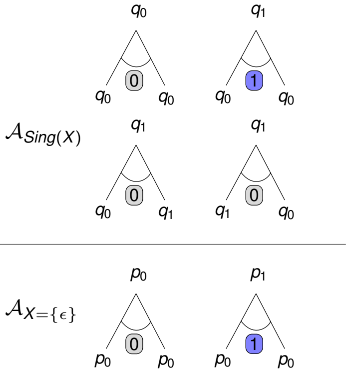
$\Delta_{\vec{0}}(\pi_X(q_0 \,\&\, p_0), \pi_X(q_0 \,\&\, p_0)) =$

$\Delta_{\vec{0}}(\pi_X(t_1), \pi_X(t_2)) \rightsquigarrow \{\pi_X(\Delta_{X=0}(t_1, t_2)), \pi_X(\Delta_{X=1}(t_1, t_2))\}$

$\{\pi_X(\Delta_{X=1}(q_0 \,\&\, p_0, q_0 \,\&\, p_0)), \pi_X(\Delta_{X=0}(\ldots))\} =$

$\Delta_a(t_1 \,\&\, t_2, t_3 \,\&\, t_4) \rightsquigarrow \Delta_a(t_1, t_2)[\&]\Delta_a(t_3, t_4)$

$\{\pi_X(\Delta_{X=1}(q_0, q_0) \,\&\, \Delta_{X=1}(p_0, p_0)), \ \ldots\} =$



Fixpoint Evaluation *ex.*

- Formula $\varphi \equiv \neg\exists X.\ \overline{Sing(X) \wedge X = \{\epsilon\}}$ and its automata term

$$t_\varphi = \left\{ \{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* \right\}$$

1. $\{ \pi_X(q_0 \,\&\, p_0) \} - \vec{0}^* =$
   - $\Delta_{\vec{0}}(\pi_X(q_0 \,\&\, p_0) , \pi_X(q_0 \,\&\, p_0)) = ?$

$\Delta_{\vec{0}}(\pi_X(q_0 \,\&\, p_0), \pi_X(q_0 \,\&\, p_0)) =$

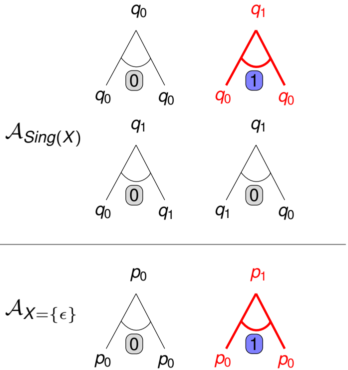
$\Delta_{\vec{0}}(\pi_X(t_1), \pi_X(t_2)) \rightsquigarrow \{\pi_X(\Delta_{X=0}(t_1, t_2)), \pi_X(\Delta_{X=1}(t_1, t_2))\}$

$\{\pi_X(\Delta_{X=1}(q_0 \,\&\, p_0, q_0 \,\&\, p_0)), \pi_X(\Delta_{X=0}(\dots))\} =$

$\Delta_a(t_1 \,\&\, t_2, t_3 \,\&\, t_4) \rightsquigarrow \Delta_a(t_1, t_2)[\&]\Delta_a(t_3, t_4)$

$\{\pi_X(\Delta_{X=1}(q_0, q_0) \,\&\, \Delta_{X=1}(p_0, p_0)), \dots\} =$

$\{\pi_X(q_1 \,\&\, p_1), \pi_X(q_0 \,\&\, p_0)\}$



Fixpoint Evaluation *ex.*

- Formula $\varphi \equiv \neg\exists X.\ \overline{Sing(X) \wedge X = \{\epsilon\}}$ and its automata term

$$t_\varphi = \Big\{ \{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* \Big\}$$

1. $\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* = \{\pi_X(q_1 \,\&\, p_1), \pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*$
   - $\Delta_{\vec{0}}(\pi_X(q_0 \,\&\, p_0), \pi_X(q_0 \,\&\, p_0)) = ?$

$\Delta_{\vec{0}}(\pi_X(q_0 \,\&\, p_0), \pi_X(q_0 \,\&\, p_0)) =$

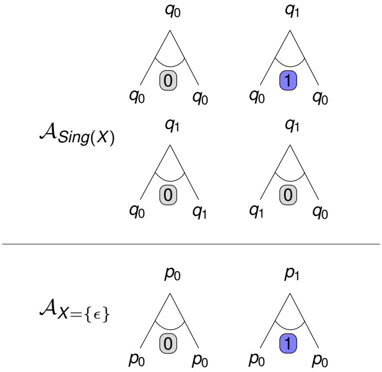
$\Delta_{\vec{0}}(\pi_X(t_1), \pi_X(t_2)) \rightsquigarrow \{\pi_X(\Delta_{X=0}(t_1, t_2)), \pi_X(\Delta_{X=1}(t_1, t_2))\}$

$\{\pi_X(\Delta_{X=1}(q_0 \,\&\, p_0, q_0 \,\&\, p_0)), \pi_X(\Delta_{X=0}(\ldots))\} =$

$\Delta_a(t_1 \,\&\, t_2, t_3 \,\&\, t_4) \rightsquigarrow \Delta_a(t_1, t_2)[\&]\Delta_a(t_3, t_4)$

$\{\pi_X(\Delta_{X=1}(q_0, q_0) \,\&\, \Delta_{X=1}(p_0, p_0)), \ldots\} =$

$\{\pi_X(q_1 \,\&\, p_1), \pi_X(q_0 \,\&\, p_0)\}$



Fixpoint Evaluation *ex.*

- Formula $\varphi \equiv \neg\exists X.\ \overline{Sing(X) \wedge X = \{\epsilon\}}$ and its automata term

$$t_\varphi = \left\{ \{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* \right\}$$

1. $\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* = \{\pi_X(q_1 \,\&\, p_1), \pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*$
   - $\Delta_{\vec{0}}(\pi_X(q_0 \,\&\, p_0), \pi_X(q_0 \,\&\, p_0)) = ?$

$$\Delta_{\vec{0}}(\pi_X(q_0 \,\&\, p_0), \pi_X(q_0 \,\&\, p_0)) =$$

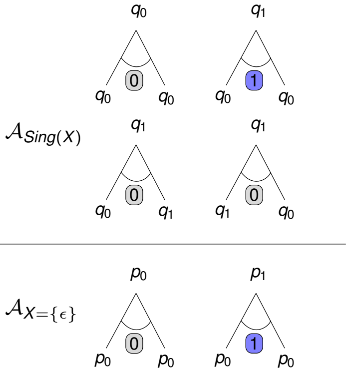
$\Delta_{\vec{0}}(\pi_X(t_1), \pi_X(t_2)) \rightsquigarrow \{\pi_X(\Delta_{X=0}(t_1, t_2)), \pi_X(\Delta_{X=1}(t_1, t_2))\}$

$$\{\pi_X(\Delta_{X=1}(q_0 \,\&\, p_0, q_0 \,\&\, p_0)), \pi_X(\Delta_{X=0}(\dots))\} =$$

$\Delta_a(t_1 \,\&\, t_2, t_3 \,\&\, t_4) \rightsquigarrow \Delta_a(t_1, t_2)[\&]\Delta_a(t_3, t_4)$

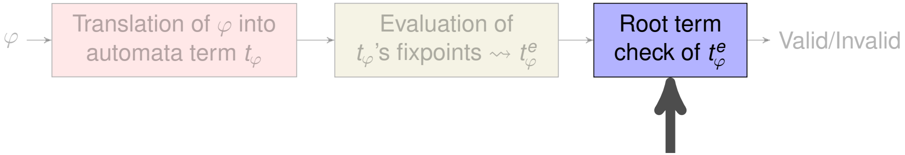
$$\{\pi_X(\Delta_{X=1}(q_0, q_0) \,\&\, \Delta_{X=1}(p_0, p_0)), \dots\} =$$

$$\{\pi_X(q_1 \,\&\, p_1), \pi_X(q_0 \,\&\, p_0)\}$$



2. ...

Overview


φ
Translation of φ into automata term t_φ
Evaluation of t_φ's fixpoints $\rightsquigarrow t_\varphi^e$
Root term check of t_φ^e
Valid/Invalid

Root Term Check

- For a ground formula φ

$$\models \varphi \iff \mathcal{L}(\mathcal{A}_\varphi) \neq \emptyset \iff \mathcal{R}(t^{\mathsf{e}}_\varphi)$$

  - $t^{\mathsf{e}}_\varphi \rightsquigarrow$ term of φ with all evaluated fixpoints

Root Term Check

- For a ground formula φ

$$\models \varphi \iff \mathcal{L}(\mathcal{A}_\varphi) \neq \emptyset \iff \mathcal{R}(t^{\mathrm{e}}_\varphi)$$

  - $t^{\mathrm{e}}_\varphi \rightsquigarrow$ term of φ with all evaluated fixpoints

- Root term check $\rightsquigarrow$ check that the corresponding TA accepts $\perp$
  - $\mathcal{R}(t \,\&\, u) \rightsquigarrow \mathcal{R}(t)$ *and* $\mathcal{R}(u)$
  - $\mathcal{R}(\pi_X(t)) \rightsquigarrow \mathcal{R}(t)$
  - $\mathcal{R}(\bar{t}) \rightsquigarrow$ *not* $\mathcal{R}(t)$
  - $\mathcal{R}(S) \rightsquigarrow$ *exists* $t \in S$ s.t. $\mathcal{R}(t)$
  - $\mathcal{R}(q) \rightsquigarrow q$ is a root state of a base TA

Efficient Decision Procedure

- Lazy evaluation
  - Driven by the root term check
  - Short-circuiting $\rightsquigarrow$ if *not* $\mathcal{R}(t_1)$, then *not* $\mathcal{R}(t_1 \,\&\, t_2)$ $\rightsquigarrow$ no need to evaluate potentially complex term t_2.
  - Early termination $\rightsquigarrow$ root term check after each step of fixpoint eval
    - Reduces number of fixpoint evaluation steps

Efficient Decision Procedure

- Lazy evaluation
  - Driven by the root term check
  - Short-circuiting $\rightsquigarrow$ if *not* $\mathcal{R}(t_1)$, then *not* $\mathcal{R}(t_1 \,\&\, t_2)$ $\rightsquigarrow$ no need to evaluate potentially complex term t_2.
  - Early termination $\rightsquigarrow$ root term check after each step of fixpoint eval
    - Reduces number of fixpoint evaluation steps

- Subsumption
  - State space pruning
  - Remove subsumed terms from a set
    - Generalization of antichain algorithm
  - $\{\{q\}, \{q, r\}\} \rightsquigarrow \{\{q, r\}\}$ (since $\mathcal{L}(\{q\}) \subseteq \mathcal{L}(\{q, r\})$)

Efficient Decision Procedure

- Lazy evaluation
  - Driven by the root term check
  - Short-circuiting $\rightsquigarrow$ if *not* $\mathcal{R}(t_1)$, then *not* $\mathcal{R}(t_1 \,\&\, t_2)$ $\rightsquigarrow$ no need to evaluate potentially complex term t_2.
  - Early termination $\rightsquigarrow$ root term check after each step of fixpoint eval
    - Reduces number of fixpoint evaluation steps

- Subsumption
  - State space pruning
  - Remove subsumed terms from a set
    - Generalization of antichain algorithm
  - $\{\{q\}, \{q, r\}\} \rightsquigarrow \{\{q, r\}\}$ (since $\mathcal{L}(\{q\}) \subseteq \mathcal{L}(\{q, r\})$)

- Product flattening
  - $\{t_1, t_2\} \,\&\, \{t_3, t_4\} \rightsquigarrow \{t_1 \,\&\, t_3, t_1 \,\&\, t_4, t_2 \,\&\, t_3, t_2 \,\&\, t_4\}$
  - Reduces size of fixpoint eval. $\rightsquigarrow$ exponential vs. polynomial size
    - E.g., $\mathcal{O}(2^{|Q_1|+|Q_2|})$ vs. $\mathcal{O}(|Q_1| \cdot |Q_2|)$

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\, \mathit{Sing}(X) \wedge X = \{\epsilon\}$ and its automata term $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{*}} \right\}$
- Is φ satisfiable?

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\, \mathit{Sing}(X) \wedge X = \{\epsilon\}$ and its automata term $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{*}} \right\}$
- Is φ satisfiable?

Root states
$\mathcal{A}_{\mathit{Sing}(X)}$: q_1
$\mathcal{A}_{X=\{\epsilon\}}$: p_1

- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{*}}\right)$

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\ \mathit{Sing}(X) \wedge X = \{\epsilon\}$ and its automata term $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{*}} \right\}$
- Is φ satisfiable?

Root states
$\mathcal{A}_{\mathit{Sing}(X)}$: q_1
$\mathcal{A}_{X=\{\epsilon\}}$: p_1

- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{*}}\right) \iff$ *not* $\mathcal{R}(\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{*})$

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\ Sing(X) \land X = \{\epsilon\}$ and its automata term $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{*}} \right\}$
- Is φ satisfiable?

Root states
$\mathcal{A}_{Sing(X)}$: q_1
$\mathcal{A}_{X=\{\epsilon\}}$: p_1

- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{*}}\right) \iff$ *not* $\mathcal{R}(\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{*})$
  - $\mathcal{R}(\pi_X(q_0 \,\&\, p_0))$

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\, Sing(X) \wedge X = \{\epsilon\}$ and its automata term $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{*}} \right\}$
- Is φ satisfiable?

Root states
$\mathcal{A}_{Sing(X)}$: q_1
$\mathcal{A}_{X=\{\epsilon\}}$: p_1

- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{*}}\right) \iff$ *not* $\mathcal{R}(\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^{*})$
  - $\mathcal{R}(\pi_X(q_0 \,\&\, p_0)) \iff \mathcal{R}(q_0 \,\&\, p_0)$

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\ \mathit{Sing}(X) \land X = \{\epsilon\}$ and its automata term $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*} \right\}$
- Is φ satisfiable?

Root states
$\mathcal{A}_{\mathit{Sing}(X)}$: q_1
$\mathcal{A}_{X=\{\epsilon\}}$: p_1

- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff$ *not* $\mathcal{R}(\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_0 \,\&\, p_0)) \iff \mathcal{R}(q_0 \,\&\, p_0) \iff \mathcal{R}(q_0)$ and $\mathcal{R}(p_0)$

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\, \mathit{Sing}(X) \land X = \{\epsilon\}$ and its automata term $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*} \right\}$
- Is φ satisfiable?

Root states
$\mathcal{A}_{\mathit{Sing}(X)}$: q_1
$\mathcal{A}_{X=\{\epsilon\}}$: p_1

- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff$ *not* $\mathcal{R}(\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_0 \,\&\, p_0)) \iff \mathcal{R}(q_0 \,\&\, p_0) \iff \mathcal{R}(q_0)$ and $\mathcal{R}(p_0) \Leftrightarrow$ *false* ✘

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\, \mathit{Sing}(X) \wedge X = \{\epsilon\}$ and its automata term $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*} \right\}$
- Is φ satisfiable?

Root states
$\mathcal{A}_{\mathit{Sing}(X)}$: q_1
$\mathcal{A}_{X=\{\epsilon\}}$: p_1

- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff$ *not* $\mathcal{R}(\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_0 \,\&\, p_0)) \iff \mathcal{R}(q_0 \,\&\, p_0) \iff \mathcal{R}(q_0)$ and $\mathcal{R}(p_0) \Leftrightarrow$ *false* ✘
  - Fixpoint eval step
    $\rightsquigarrow \{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* = \{\pi_X(q_0 \,\&\, p_0), \pi_X(q_1 \,\&\, p_1)\} - \vec{0}^*$

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\ Sing(X) \wedge X = \{\epsilon\}$ and its automata term
  $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*} \right\}$
- Is φ satisfiable?

Root states
$\mathcal{A}_{Sing(X)}$: q_1
$\mathcal{A}_{X=\{\epsilon\}}$: p_1

- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff$ *not* $\mathcal{R}(\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_0 \,\&\, p_0)) \iff \mathcal{R}(q_0 \,\&\, p_0) \iff \mathcal{R}(q_0)$ and $\mathcal{R}(p_0) \Leftrightarrow$ *false* ✘
  - Fixpoint eval step
    $\rightsquigarrow \{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* = \{\pi_X(q_0 \,\&\, p_0), \pi_X(q_1 \,\&\, p_1)\} - \vec{0}^*$
- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff$ *not* $\mathcal{R}(\{\pi_X(q_0 \,\&\, p_0), \pi_X(q_1 \,\&\, p_1)\} - \vec{0}^*)$

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\, Sing(X) \land X = \{\epsilon\}$ and its automata term $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*} \right\}$
- Is φ satisfiable?

> Root states
> $\mathcal{A}_{Sing(X)}$: q_1
> $\mathcal{A}_{X=\{\epsilon\}}$: p_1

- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff not\ \mathcal{R}(\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_0 \,\&\, p_0)) \iff \mathcal{R}(q_0 \,\&\, p_0) \iff \mathcal{R}(q_0)$ and $\mathcal{R}(p_0) \Leftrightarrow$ *false* ✘
  - Fixpoint eval step
    $\rightsquigarrow \{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* = \{\pi_X(q_0 \,\&\, p_0), \pi_X(q_1 \,\&\, p_1)\} - \vec{0}^*$
- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff not\ \mathcal{R}(\{\pi_X(q_0 \,\&\, p_0), \pi_X(q_1 \,\&\, p_1)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_1 \,\&\, p_1))$

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\ Sing(X) \wedge X = \{\epsilon\}$ and its automata term
  $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*} \right\}$
- Is φ satisfiable?

Root states
$\mathcal{A}_{Sing(X)}$: q_1
$\mathcal{A}_{X=\{\epsilon\}}$: p_1

- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff$ *not* $\mathcal{R}(\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_0 \,\&\, p_0)) \iff \mathcal{R}(q_0 \,\&\, p_0) \iff \mathcal{R}(q_0)$ and $\mathcal{R}(p_0) \Leftrightarrow$ *false* ✘
  - Fixpoint eval step
    $\rightsquigarrow \{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* = \{\pi_X(q_0 \,\&\, p_0), \pi_X(q_1 \,\&\, p_1)\} - \vec{0}^*$
- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff$ *not* $\mathcal{R}(\{\pi_X(q_0 \,\&\, p_0), \pi_X(q_1 \,\&\, p_1)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_1 \,\&\, p_1)) \iff \mathcal{R}(q_1 \,\&\, p_1)$

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\ Sing(X) \land X = \{\epsilon\}$ and its automata term $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*} \right\}$
- Is φ satisfiable?

Root states
$\mathcal{A}_{Sing(X)}$: q_1
$\mathcal{A}_{X=\{\epsilon\}}$: p_1

- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff \textit{not}\ \mathcal{R}(\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_0 \,\&\, p_0)) \iff \mathcal{R}(q_0 \,\&\, p_0) \iff \mathcal{R}(q_0)$ and $\mathcal{R}(p_0) \Leftrightarrow$ *false* ✘
  - Fixpoint eval step
    $\rightsquigarrow \{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* = \{\pi_X(q_0 \,\&\, p_0), \pi_X(q_1 \,\&\, p_1)\} - \vec{0}^*$
- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff \textit{not}\ \mathcal{R}(\{\pi_X(q_0 \,\&\, p_0), \pi_X(q_1 \,\&\, p_1)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_1 \,\&\, p_1)) \iff \mathcal{R}(q_1 \,\&\, p_1) \iff \mathcal{R}(q_1)$ and $\mathcal{R}(p_1)$

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\, Sing(X) \land X = \{\epsilon\}$ and its automata term $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*} \right\}$
- Is φ satisfiable?

Root states
$\mathcal{A}_{Sing(X)}$: q_1
$\mathcal{A}_{X=\{\epsilon\}}$: p_1

- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff \textit{not}\ \mathcal{R}(\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_0 \,\&\, p_0)) \iff \mathcal{R}(q_0 \,\&\, p_0) \iff \mathcal{R}(q_0)$ and $\mathcal{R}(p_0) \Leftrightarrow$ *false* ✘
  - Fixpoint eval step
    $\rightsquigarrow \{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* = \{\pi_X(q_0 \,\&\, p_0), \pi_X(q_1 \,\&\, p_1)\} - \vec{0}^*$
- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff \textit{not}\ \mathcal{R}(\{\pi_X(q_0 \,\&\, p_0), \pi_X(q_1 \,\&\, p_1)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_1 \,\&\, p_1)) \iff \mathcal{R}(q_1 \,\&\, p_1) \iff \mathcal{R}(q_1)$ and $\mathcal{R}(p_1) \Leftrightarrow$ *true* ✔

Lazy Evaluation

- Formula $\varphi \equiv \neg\exists X.\ \mathit{Sing}(X) \wedge X = \{\epsilon\}$ and its automata term $t_\varphi = \left\{ \overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*} \right\}$
- Is φ satisfiable?

Root states
$\mathcal{A}_{\mathit{Sing}(X)}$: q_1
$\mathcal{A}_{X=\{\epsilon\}}$: p_1

- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff \mathit{not}\ \mathcal{R}(\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_0 \,\&\, p_0)) \iff \mathcal{R}(q_0 \,\&\, p_0) \iff \mathcal{R}(q_0)$ and $\mathcal{R}(p_0) \Leftrightarrow$ *false* ✘
  - Fixpoint eval step $\rightsquigarrow \{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^* = \{\pi_X(q_0 \,\&\, p_0), \pi_X(q_1 \,\&\, p_1)\} - \vec{0}^*$
- $\mathcal{R}\left(\overline{\{\pi_X(q_0 \,\&\, p_0)\} - \vec{0}^*}\right) \iff \mathit{not}\ \mathcal{R}(\{\pi_X(q_0 \,\&\, p_0), \pi_X(q_1 \,\&\, p_1)\} - \vec{0}^*)$
  - $\mathcal{R}(\pi_X(q_1 \,\&\, p_1)) \iff \mathcal{R}(q_1 \,\&\, p_1) \iff \mathcal{R}(q_1)$ and $\mathcal{R}(p_1) \Leftrightarrow$ *true* ✔
- $\mathcal{R}(t_\varphi)$ is *false* $\rightsquigarrow$ formula is unsatisfiable

Experiments

- Prototype tool written in Haskell
- Comparison with MONA (highly optimised C++)
  - MONA usually quite faster
  - some formulae where MONA was much slower (see below)

1 $\varphi_n^{pt} \equiv \forall Z_1, Z_2.\ \exists X_1, \ldots, X_n.\ edge(Z_1, X_1) \wedge \bigwedge_{i=1}^{n} edge(X_i, X_{i+1}) \wedge edge(X_n, Z_2)$
where

- $edge(X, Y) \equiv edge_{\mathtt{L}}(X, Y) \vee edge_{\mathtt{R}}(X, Y)$
- $edge_{\mathtt{L}/\mathtt{R}}(X, Y) \equiv \exists Z.\ Z = S_{\mathtt{L}/\mathtt{R}}(X) \wedge Z \subseteq Y$

	running time (sec)		# of subterms/states	
n	*Lazy*	MONA	*Lazy*	MONA
1	0.02	0.16	149	216
2	0.50	–	937	–
3	0.83	–	2,487	–
4	34.95	–	8,391	–
5	60.94	–	23,827	–

Experiments

2 $\varphi_n^{cnst} \equiv \exists X.\ X = \{(\mathrm{LR})^4\} \wedge X = \{(\mathrm{LR})^n\}$.

	running time (sec)		# of subterms/states	
n	*Lazy*	MONA	*Lazy*	MONA
80	14.60	40.07	1,146	27,913
90	21.03	64.26	1,286	32,308
100	28.57	98.42	1,426	36,258
110	38.10	–	1,566	–
120	49.82	–	1,706	–

3 $\varphi_n^{sub} = \forall X_1, \ldots, X_n\ \exists Y.\ \bigwedge_{i=1}^{n-1} X_i \subseteq Y \Rightarrow (X_{i+1} = S_{\mathrm{L}}(Y) \vee X_{i+1} = S_{\mathrm{R}}(Y))$.

	running time (sec)		# of subterms/states	
n	*Lazy*	MONA	*Lazy*	MONA
3	0.01	0.00	140	92
4	0.04	34.39	386	170
5	0.24	–	981	–
6	2.01	–	2,376	–

Conclusion and Future Work

- Representation of constructed automata using automata terms
- Efficient decision procedure
  - Lazy evaluation of automata terms directed by root term check
  - Subsumption
  - Product flattening
- New line of attack on hard WS2S formulae!

- Future work
  - Formula preprocessing (antiprenexing, . . .)

Conclusion and Future Work

- Representation of constructed automata using automata terms
- Efficient decision procedure
  - Lazy evaluation of automata terms directed by root term check
  - Subsumption
  - Product flattening
- New line of attack on hard WS2S formulae!

- Future work
  - Formula preprocessing (antiprenexing, . . .)

THANK YOU