Tree Automata, Approximations, and Constraints for Verification

Ph.D. thesis defence for **Vincent Hugot**,
Supervised by **O. Kouchnarenko** and **P.-C. Héam**

{pheam,vhugot,okouchna}@femto-st.fr

University of Franche-Comté
DGA & Inria/CASSIS & FEMTO-ST (DISC)

August 1, 2014

Model-Checking

Introduced in
[Clarke and Emerson, 1981, Queille and Sifakis, 1982]

Check $\mathbf{M}, \mathbf{s_0} \models \boldsymbol{\varphi}$:
"do all executions of M starting in s_0 follow φ?"

$\mathbf{M}$	finite states/transitions model
$\mathbf{s_0}$	initial state
$\boldsymbol{\varphi}$	the specification, in temporal logic

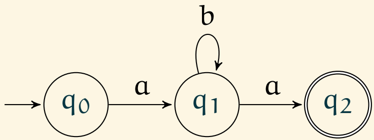
Limited by **state explosion**. Prevented by **parametrisation**.

Regular Model-Checking

Introduced in [Kesten et al., 1997]

regular model-checking.

states →	finite words
sets of states →	finite-state automata
transitions →	finite-state transducers, semi-Thue systems



$\leftrightarrow \quad \{aa, aba, abba, \dots\}$

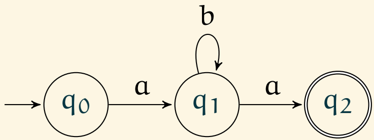
Automata provide finite, tractable symbolic representations of **infinite sets** of states.

Regular Model-Checking

Introduced in [Kesten et al., 1997]

tree regular model-checking.

states $\rightarrow$	**finite trees**
sets of states $\rightarrow$	tree automata
transitions $\rightarrow$	tree transducers, term rewriting systems



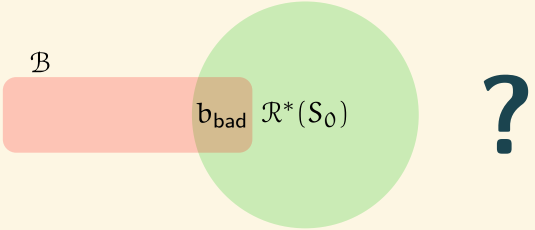
$\leftrightarrow \quad \{aa, aba, abba, \dots\}$

Automata provide finite, tractable symbolic representations of **infinite sets** of states.

Reachability Analysis (in TRMC)

e.g. [Feuillade et al., 2004, Bouajjani and Touili, 2002]

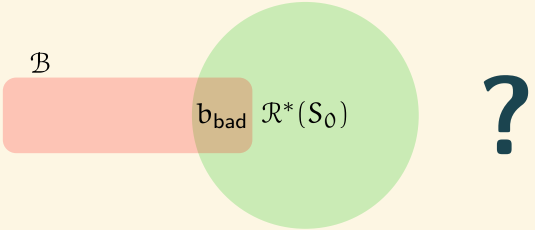
$\mathcal{S}_0$	initial language	*tree automaton*
$\mathcal{B}$	set of "bad" states	*tree automaton*
$\mathcal{R}$	the transitions	*rewrite system or transducer*



Reachability Analysis (in TRMC)

e.g. [Feuillade et al., 2004, Bouajjani and Touili, 2002]

S_0	initial language	*tree automaton*
$\mathcal{B}$	set of "bad" states	*tree automaton*
$\mathcal{R}$	the transitions	*rewrite system or transducer*

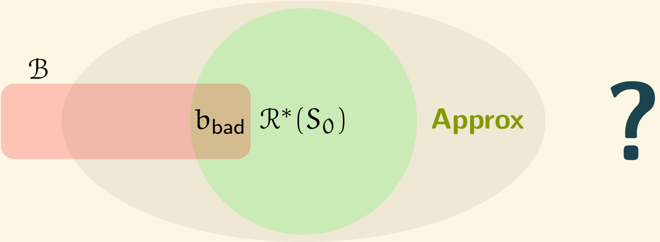


- Regularity-preserving **classes**, context-free step,...

Reachability Analysis (in TRMC)

e.g. [Feuillade et al., 2004, Bouajjani and Touili, 2002]

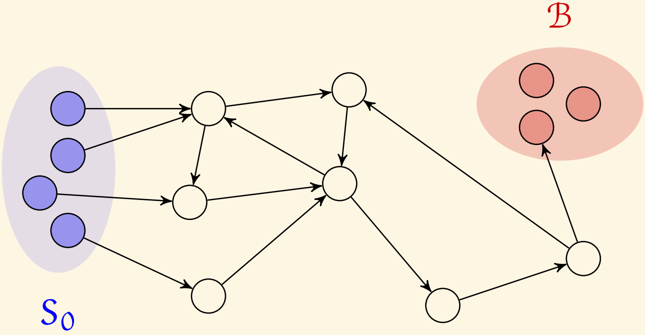
$\mathcal{S}_0$	initial language	*tree automaton*
$\mathfrak{B}$	set of "bad" states	*tree automaton*
$\mathfrak{R}$	the transitions	*rewrite system or transducer*



- Regularity-preserving **classes**, context-free step,. . .
- Regular over- or under-**approximations**.

Variations on Reachability Analysis

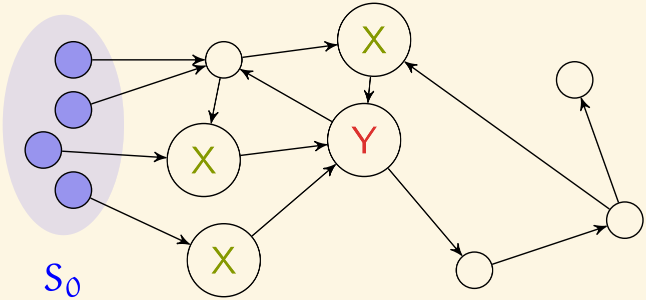
With Rewriting: e.g. [Meseguer, 1992, Boyer and Genet, 2009, Courbis et al., 2009]



Reachability analysis = $\Box\neg\mathcal{B}$.

Variations on Reachability Analysis

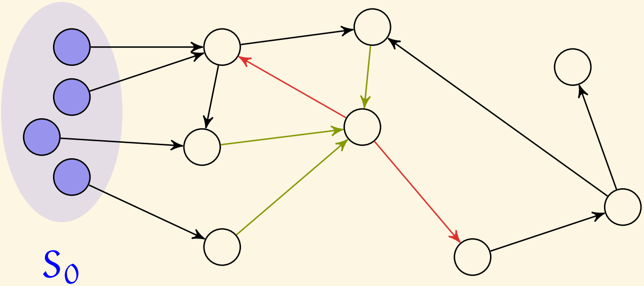
With Rewriting: e.g.
[Meseguer, 1992, Boyer and Genet, 2009, Courbis et al., 2009]



Reachability analysis $= \Box \neg \mathcal{B}$. More general: e.g. $\Box(X \Rightarrow \circ Y)$.

Variations on Reachability Analysis

With Rewriting: e.g. [Meseguer, 1992, Boyer and Genet, 2009, Courbis et al., 2009]



Reachability analysis = $\square \neg \mathcal{B}$. More general: e.g. $\square(X \Rightarrow \circ Y)$.
Same on transitions: $\square(♠ \Rightarrow \circ ♠)$.

1. Model-Checking LTL on Rewrite Sequences
   - Statement of the Central Problem
   - Our Approach: An Overview

2. TAGE With a Bounded Number of Constraints
   - Global Equality Constraints
   - Overview of the Results

3. Other Works and Some Perspectives
   - Results on SAT & Tree-Walking Automata
   - Perspectives and Questions

Model-Checking Rewrite Sequences

[Meseguer, 1992]

Order of application of rewrite rules.

Check $\mathcal{R}, \Pi \models \varphi$, with

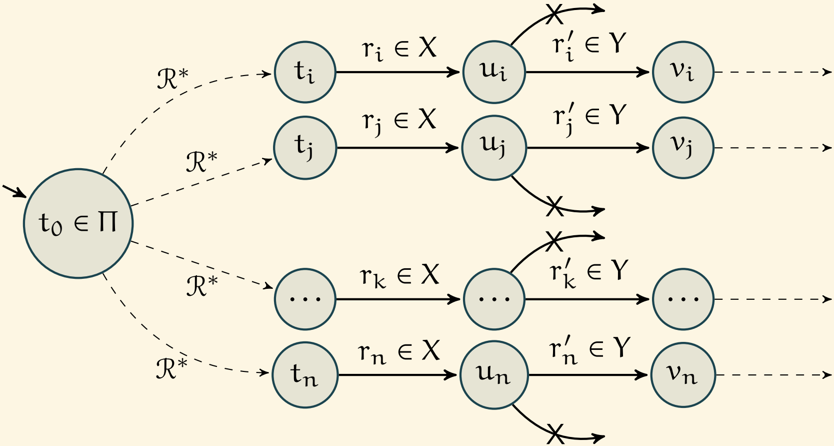
$\mathcal{R}$	a term rewriting system (TRS)
Π	the initial (regular) tree language
φ	a linear temporal logic (LTL) formula

Example: $\varphi = \square(X \Rightarrow \bullet Y)$

$X, Y \subseteq \mathcal{R}$ are sets of rules

Model-Checking Rewrite Sequences

[Meseguer, 1992]



$\varphi = \Box(X \Rightarrow \bullet Y)$

Model-Checking Rewrite Sequences

[Meseguer, 1992]

Order of application of rewrite rules.

Check $\mathcal{R}, \Pi \models \varphi$, with

$\mathcal{R}$	a term rewriting system (TRS)
Π	the initial (regular) tree language
φ	a linear temporal logic (LTL) formula

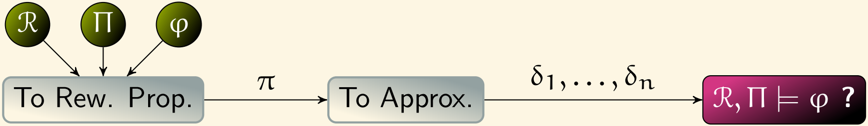
Example: $\varphi = \square(X \Rightarrow \bullet Y)$

$X, Y \subseteq \mathcal{R}$ are sets of rules
$X =$ "ask PIN code" $= \{\mathsf{ask}\}$
$Y =$ "authenticate or cancel" $= \{\mathsf{auth}_1, \mathsf{auth}_2, \mathsf{can}\}$

Model-Checking Rewrite Sequences

Overview of the Model-Checking Process

Whether $\mathcal{R}, \Pi \models \varphi$ is **undecidable**.



Two-step **positive approximated decision** [Courbis et al., 2009]:

π	a rewrite proposition – language equation
δ_k	TAGE-based approximated procedures
TAGE	tree automata with constraints: more precision

Model-Checking Rewrite Sequences

Prior work [Courbis et al., 2009]

"The system $\mathcal{R}$ **satisfies the property**"...

$$\mathcal{R}, \Pi \models \ \square(X \Rightarrow \bullet Y)$$

... is equivalent to the **rewrite proposition**...

$$[\mathcal{R} \setminus Y](X(\mathcal{R}^*(\Pi))) = \varnothing \ \wedge \ X(\mathcal{R}^*(\Pi)) \subseteq Y^{-1}(\mathcal{T})$$

Model-Checking Rewrite Sequences

Prior work [Courbis et al., 2009]

"The system $\mathcal{R}$ **satisfies the property**"...

$$\mathcal{R}, \Pi \models \ \Box(X \Rightarrow \bullet Y)$$

... is equivalent to the **rewrite proposition**...

$$[\mathcal{R} \setminus Y](X(\mathcal{R}^*(\Pi))) = \varnothing \ \wedge \ X(\mathcal{R}^*(\Pi)) \subseteq Y^{-1}(\mathcal{T})$$

... **approximated** with TAGE by, assuming Y is left-linear,

$\mathtt{IsEmpty}(\mathtt{OneStep}(\mathcal{R} \setminus Y, \mathtt{Approx}(\mathcal{A}, \mathcal{R})), X)$ and
$\mathtt{Subset}(\mathtt{OneStep}(X, \mathtt{Approx}(\mathcal{A}, \mathcal{R})), \mathtt{Backward}(Y))$,
where $\mathcal{L}(\mathcal{A}) = \Pi$, $\mathcal{L}(\mathtt{Approx}(\mathcal{A}, \mathcal{R})) \supseteq \mathcal{R}^*(\mathcal{L}(\mathcal{A}))$

Model-Checking Rewrite Sequences

Prior work [Courbis et al., 2009], and New Goals

1. $\mathcal{R}, \Pi \models \ \square(X \Rightarrow \bullet Y)$

$$[\mathcal{R} \setminus Y](X(\mathcal{R}^*(\Pi))) = \varnothing \ \wedge \ X(\mathcal{R}^*(\Pi)) \subseteq Y^{-1}(\mathcal{T})$$

2. $\mathcal{R}, \Pi \models \ \neg Y \ \wedge \ \square(\bullet Y \Rightarrow X)$

$$Y(\Pi) = \varnothing \ \wedge \ Y([\mathcal{R} \setminus X](\mathcal{R}^*(\Pi))) = \varnothing$$

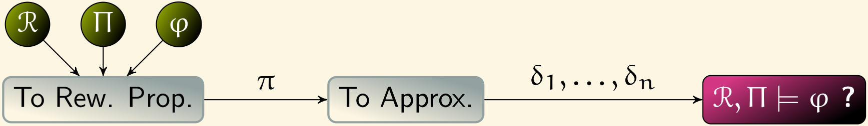
3. $\mathcal{R}, \Pi \models \ \square(X \Rightarrow \circ \square \neg Y)$

$$Y(\mathcal{R}^*(X(\mathcal{R}^*(\Pi)))) = \varnothing$$

Model-Checking Rewrite Sequences

Prior work [Courbis et al., 2009], and New Goals

Main goal: from **manual** to **automatic** translations.



Sub-goal: **efficient** procedures $\implies$ TAGE complexity study.

1. Model-Checking LTL on Rewrite Sequences
   - Statement of the Central Problem
   - Our Approach: An Overview

2. TAGE With a Bounded Number of Constraints
   - Global Equality Constraints
   - Overview of the Results

3. Other Works and Some Perspectives
   - Results on SAT & Tree-Walking Automata
   - Perspectives and Questions

Intuitions for the Translation

1. $\mathcal{R}, \Pi \models \neg \mathbf{X}$:

"The first transition, if it occurs, is not by X"

$$\pi_1 \equiv X(\Pi) = \varnothing$$

Intuitions for the Translation

1. $\neg X$:

"The first transition, if it occurs, is not by X"

$$\pi_1 \equiv X(\Pi) = \varnothing$$

2. X:

"There is a first transition, and it is by X"

$$\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \textbf{ ?}$$

Intuitions for the Translation

1. $\neg X$:

"The first transition, if it occurs, is not by X"

$$\pi_1 \;\equiv\; X(\Pi) = \varnothing$$

2. X:

"There is a first transition, and it is by X"

$$\pi_2 \;\equiv\; [\mathcal{R} \setminus X](\Pi) = \varnothing \quad\wedge\; \Pi \subseteq X^{-1}(\mathcal{T})$$

Intuitions for the Translation

1. $\neg X$:

"The first transition, if it occurs, is not by X"

$$\pi_1 \equiv X(\Pi) = \varnothing$$

2. X:

"There is a first transition, and it is by X"

$$\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \quad \wedge \; \Pi \subseteq X^{-1}(\mathcal{T})$$

3. $\square \neg X$:

"No transition that occurs is by X"

$$\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing$$

Intuitions for the Translation

1. **¬X:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \;\wedge\; \Pi \subseteq X^{-1}(\mathcal{T})$
3. **$\square\neg$X:**

"No transition that occurs is by X"

$$\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$$

4. **$\square$X:**

"All transitions that occur are by X"

Intuitions for the Translation

1. **¬X:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \;\wedge\; \Pi \subseteq X^{-1}(\mathcal{T})$
3. **□¬X:**

"No transition that occurs is by X"

$$\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$$

4. **□X:**

"All transitions that occur are by X"

$$\pi_4 \equiv \pi_2[\mathcal{R}^*(\Pi)/\Pi]$$
$$\equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing \;\wedge\; \mathcal{R}^*(\Pi) \subseteq X^{-1}(\mathcal{T})$$

?

Intuitions for the Translation

1. **¬X:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \;\wedge\; \Pi \subseteq X^{-1}(\mathcal{T})$
3. **□¬X:**

"No transition that occurs is by X"

$$\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$$

4. **□X:**

"All transitions that occur are by X"

$$\pi_4 \equiv \pi_2[\mathcal{R}^*(\Pi)/\Pi]$$
$$\equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing \;\wedge\; \mathcal{R}^*(\Pi) \subseteq X^{-1}(\mathcal{T})$$

?

ω-language! Too strong

Intuitions for the Translation

1. **¬X:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \quad \wedge \quad \Pi \subseteq X^{-1}(\mathcal{T})$
3. **$\Box$¬X:**

"No transition that occurs is by X"

$$\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$$

4. **$\Box$X:**

"All transitions that occur are by X"

$$\pi_4 \equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing$$

Intuitions for the Translation

1. **¬X:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \quad \wedge \; \Pi \subseteq X^{-1}(\mathcal{T})$
3. **□¬X:** $\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$
4. **□X:** $\pi_4 \equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing$
5. **Conjunction:** if $\varphi : \pi_5$ and $\psi : \pi'_5$ then $\varphi \wedge \psi : \pi_5 \wedge \pi'_5$.

$\varphi : \pi$ = "$\mathcal{R}, \Pi \models \varphi$ is translated by π"
"for all executions, φ is satisfied"

$$\forall x.P(x) \;\wedge\; \forall x.Q(x) \iff \forall x.(P(x) \;\wedge\; Q(x))$$

Intuitions for the Translation

1. **¬X:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \;\wedge\; \Pi \subseteq X^{-1}(\mathcal{T})$
3. **$\square$ ¬X:** $\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$
4. **$\square$ X:** $\pi_4 \equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing$
5. **Conjunction:** if $\varphi : \pi_5$ and $\psi : \pi_5'$ then $\varphi \wedge \psi : \pi_5 \wedge \pi_5'$.

$$\forall x.P(x) \;\wedge\; \forall x.Q(x) \iff \forall x.(P(x) \;\wedge\; Q(x))$$

6. **Disjunction:** $\pi_6 \vee \pi_6' \implies \mathcal{R}, \Pi \models \varphi \vee \psi$

$$\forall x.P(x) \;\vee\; \forall x.Q(x) \implies \forall x.(P(x) \;\vee\; Q(x))$$

Intuitions for the Translation

1. **$\neg X$:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \;\wedge\; \Pi \subseteq X^{-1}(\mathcal{T})$
3. **$\Box \neg X$:** $\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$
4. **$\Box X$:** $\pi_4 \equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing$
5. **Conjunction:** if $\varphi : \pi_5$ and $\psi : \pi_5'$ then $\varphi \wedge \psi : \pi_5 \wedge \pi_5'$.

$$\forall x.P(x) \;\wedge\; \forall x.Q(x) \iff \forall x.(P(x) \;\wedge\; Q(x))$$

6. **Disjunction:** $\pi_6 \vee \pi_6' \implies \mathcal{R}, \Pi \models \varphi \vee \psi$

$$\forall x.P(x) \;\vee\; \forall x.Q(x) \implies \forall x.(P(x) \;\vee\; Q(x))$$

7. **Negation:** $\mathcal{R}, \Pi \not\models \varphi \;\neq\; \mathcal{R}, \Pi \models \neg\varphi$: “NNF” required

$$\forall x.\neg P(x) \neq \neg \forall x.P(x)$$

Intuitions for the Translation

1. **¬X:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \;\wedge\; \Pi \subseteq X^{-1}(\mathcal{T})$
3. **$\square$¬X:** $\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$
4. **$\square$X:** $\pi_4 \equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing$
5. **Conjunction:** if $\varphi : \pi_5$ and $\psi : \pi_5'$ then $\varphi \wedge \psi : \pi_5 \wedge \pi_5'$.
6. **Disjunction:** $\pi_6 \vee \pi_6' \implies \mathcal{R}, \Pi \models \varphi \vee \psi$
7. **Negation:** $\mathcal{R}, \Pi \not\models \varphi \;\neq\; \mathcal{R}, \Pi \models \neg\varphi$: “NNF” required
8. **Implication:** $X \Rightarrow \bullet Y$:

Intuitions for the Translation

1. **¬X:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \ \wedge \ \Pi \subseteq X^{-1}(\mathcal{T})$
3. **$\square$¬X:** $\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$
4. **$\square$X:** $\pi_4 \equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing$
5. **Conjunction:** if $\varphi : \pi_5$ and $\psi : \pi_5'$ then $\varphi \wedge \psi : \pi_5 \wedge \pi_5'$.
6. **Disjunction:** $\pi_6 \vee \pi_6' \implies \mathcal{R}, \Pi \models \varphi \vee \psi$
7. **Negation:** $\mathcal{R}, \Pi \not\models \varphi \ \neq \ \mathcal{R}, \Pi \models \neg\varphi$: "NNF" required
8. **Implication:** $X \Rightarrow \bullet Y$:
   $\pi_7 \equiv [\mathcal{R} \setminus Y](X(\Pi)) = \varnothing \ \wedge \ X(\Pi) \subseteq Y^{-1}(\mathcal{T})$

Intuitions for the Translation

1. **$\neg X$:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \quad \wedge \quad \Pi \subseteq X^{-1}(\mathcal{T})$
3. **$\Box \neg X$:** $\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$
4. **$\Box X$:** $\pi_4 \equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing$
5. **Conjunction:** if $\varphi : \pi_5$ and $\psi : \pi_5'$ then $\varphi \wedge \psi : \pi_5 \wedge \pi_5'$.
6. **Disjunction:** $\pi_6 \vee \pi_6' \implies \mathcal{R}, \Pi \models \varphi \vee \psi$
7. **Negation:** $\mathcal{R}, \Pi \not\models \varphi \neq \mathcal{R}, \Pi \models \neg\varphi$: "NNF" required
8. **Implication:** $X \Rightarrow \bullet Y$:
   $\pi_7 \equiv [\mathcal{R} \setminus Y](X(\Pi)) = \varnothing \wedge X(\Pi) \subseteq Y^{-1}(\mathcal{T})$
   $X : \pi_2$, $Y : \pi_2' \equiv \pi_2[Y/X]$, $\pi_7 \equiv \pi_2'[X(\Pi)/\Pi]$

Intuitions for the Translation

1. **¬X:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \quad \wedge \quad \Pi \subseteq X^{-1}(\mathcal{T})$
3. **$\Box\neg$X:** $\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$
4. **$\Box$X:** $\pi_4 \equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing$
5. **Conjunction:** if $\varphi : \pi_5$ and $\psi : \pi'_5$ then $\varphi \wedge \psi : \pi_5 \wedge \pi'_5$.
6. **Disjunction:** $\pi_6 \vee \pi'_6 \implies \mathcal{R}, \Pi \models \varphi \vee \psi$
7. **Negation:** $\mathcal{R}, \Pi \not\models \varphi \neq \mathcal{R}, \Pi \models \neg\varphi$: "NNF" required
8. **Implication:** $X \Rightarrow \bullet Y$:
   $\pi_7 \equiv [\mathcal{R} \setminus Y](X(\Pi)) = \varnothing \wedge X(\Pi) \subseteq Y^{-1}(\mathcal{T})$
   $X : \pi_2$, $Y : \pi'_2 \equiv \pi_2[Y/X]$, $\pi_7 \equiv \pi'_2[X(\Pi)/\Pi]$

Intuitions for the Translation

1. $\neg X$: $\pi_1 \equiv X(\Pi) = \varnothing$
2. X: $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \;\wedge\; \Pi \subseteq X^{-1}(\mathcal{T})$
3. $\Box\neg X$: $\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$
4. $\Box X$: $\pi_4 \equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing$
5. **Conjunction:** if $\varphi : \pi_5$ and $\psi : \pi_5'$ then $\varphi \wedge \psi : \pi_5 \wedge \pi_5'$.
6. **Disjunction:** $\pi_6 \vee \pi_6' \implies \mathcal{R}, \Pi \models \varphi \vee \psi$
7. **Negation:** $\mathcal{R}, \Pi \not\models \varphi \;\neq\; \mathcal{R}, \Pi \models \neg\varphi$: “NNF” required
8. **Implication:** $X \Rightarrow \bullet Y$:
   $\pi_7 \equiv [\mathcal{R} \setminus Y](X(\Pi)) = \varnothing \;\wedge\; X(\Pi) \subseteq Y^{-1}(\mathcal{T})$
   $X : \pi_2,\; Y : \pi_2' \equiv \pi_2[Y/X]$, $\pi_7 \equiv \pi_2'[X(\Pi)/\Pi]$

Intuitions for the Translation

1. **¬X:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \;\wedge\; \Pi \subseteq X^{-1}(\mathcal{T})$
3. **$\Box$¬X:** $\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$
4. **$\Box$X:** $\pi_4 \equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing$
5. **Conjunction:** if $\varphi : \pi_5$ and $\psi : \pi_5'$ then $\varphi \wedge \psi : \pi_5 \wedge \pi_5'$.
6. **Disjunction:** $\pi_6 \vee \pi_6' \implies \mathcal{R}, \Pi \models \varphi \vee \psi$
7. **Negation:** $\mathcal{R}, \Pi \not\models \varphi \;\neq\; \mathcal{R}, \Pi \models \neg\varphi$: "NNF" required
8. **Implication:** $X \Rightarrow \bullet Y$:
   $\pi_7 \equiv [\mathcal{R} \setminus Y](X(\Pi)) = \varnothing \;\wedge\; X(\Pi) \subseteq Y^{-1}(\mathcal{T})$
   $X : \pi_2,\ Y : \pi_2' \equiv \pi_2[Y/X],\ \pi_7 \equiv \pi_2'[X(\Pi)/\Pi]$
   $\Box(X \Rightarrow \bullet Y) : \pi_0 \equiv \pi_7[\mathcal{R}^*(\Pi)/\Pi]$

Intuitions for the Translation

1. **$\neg X$:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \;\wedge\; \Pi \subseteq X^{-1}(\mathcal{T})$
3. **$\square \neg X$:** $\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$
4. **$\square X$:** $\pi_4 \equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing$
5. **Conjunction:** if $\varphi : \pi_5$ and $\psi : \pi'_5$ then $\varphi \wedge \psi : \pi_5 \wedge \pi'_5$.
6. **Disjunction:** $\pi_6 \vee \pi'_6 \implies \mathcal{R}, \Pi \models \varphi \vee \psi$
7. **Negation:** $\mathcal{R}, \Pi \not\models \varphi \neq \mathcal{R}, \Pi \models \neg\varphi$: "NNF" required
8. **Implication:** $X \Rightarrow \bullet Y$:
   $\pi_7 \equiv [\mathcal{R} \setminus Y](X(\Pi)) = \varnothing \;\wedge\; X(\Pi) \subseteq Y^{-1}(\mathcal{T})$
   $X : \pi_2,\ Y : \pi'_2 \equiv \pi_2[Y/X],\ \pi_7 \equiv \pi'_2[X(\Pi)/\Pi]$
   $\square(X \Rightarrow \bullet Y) : \pi_0 \equiv \pi_7[\mathcal{R}^*(\Pi)/\Pi]$
   What about $\bullet Y \Rightarrow X$?

Intuitions for the Translation

1. **$\neg X$:** $\pi_1 \equiv X(\Pi) = \varnothing$
2. **X:** $\pi_2 \equiv [\mathcal{R} \setminus X](\Pi) = \varnothing \wedge \Pi \subseteq X^{-1}(\mathcal{T})$
3. **$\Box \neg X$:** $\pi_3 \equiv X(\mathcal{R}^*(\Pi)) = \varnothing \equiv \pi_1[\mathcal{R}^*(\Pi)/\Pi]$
4. **$\Box X$:** $\pi_4 \equiv [\mathcal{R} \setminus X](\mathcal{R}^*(\Pi)) = \varnothing$
5. **Conjunction:** if $\varphi : \pi_5$ and $\psi : \pi_5'$ then $\varphi \wedge \psi : \pi_5 \wedge \pi_5'$.
6. **Disjunction:** $\pi_6 \vee \pi_6' \implies \mathcal{R}, \Pi \models \varphi \vee \psi$
7. **Negation:** $\mathcal{R}, \Pi \not\models \varphi \neq \mathcal{R}, \Pi \models \neg\varphi$: "NNF" required
8. **Implication:** $X \Rightarrow \bullet Y$:
   $\pi_7 \equiv [\mathcal{R} \setminus Y](X(\Pi)) = \varnothing \wedge X(\Pi) \subseteq Y^{-1}(\mathcal{T})$
   $X : \pi_2,\ Y : \pi_2' \equiv \pi_2[Y/X],\ \pi_7 \equiv \pi_2'[X(\Pi)/\Pi]$
   $\Box(X \Rightarrow \bullet Y) : \pi_0 \equiv \pi_7[\mathcal{R}^*(\Pi)/\Pi]$
   What about $\bullet Y \Rightarrow X$? Other techniques (signatures,...)

Translation Rules, by Examples

A dozen rules, e.g. **conjunction:**

$$\updownarrow \frac{\langle \Pi \,⨾\, \sigma \Vdash \varphi \wedge \psi \rangle}{\langle \Pi \,⨾\, \sigma \Vdash \varphi \rangle \;\wedge\; \langle \Pi \,⨾\, \sigma \Vdash \psi \rangle}$$

always (simplest case):

$$\updownarrow \frac{\langle \Pi \,⨾\, \varepsilon \Vdash \square \varphi \rangle}{\langle \mathcal{R}^*(\Pi) \,⨾\, \star\varepsilon \Vdash \varphi \rangle}$$

positive literal:

$$\updownarrow \frac{\langle \Pi \,⨾\, \sigma \Vdash X \rangle \qquad (\sigma \setminus X) \lhd \hbar(\sigma \setminus X) = \varepsilon}{\Pi_{\sigma\setminus X}^{\hbar(\sigma\setminus X)} = \varnothing \quad \wedge \bigwedge_{k \in \nabla\sigma, k=0}^{\hbar(\sigma\setminus X)-1} \Pi_{\sigma\setminus X}^{k} \subseteq \mathcal{R}^{-1}(\mathcal{T})}$$

LTL → Rewrite Proposition

Derivation Tree

Derivation tree: automatic translation and proof

$$\updownarrow \frac{\langle \Pi \fatsemi \varepsilon \Vdash \Box(X \Rightarrow \bullet Y)\rangle}{\updownarrow \frac{\langle \mathcal{R}^*(\Pi) \fatsemi \star\varepsilon \Vdash X \Rightarrow \bullet Y\rangle}{\updownarrow \frac{\langle \mathcal{R}^*(\Pi) \fatsemi \wr X \fatsemi \mathcal{R} \mid \overline{\mathbb{N}}_1 \int \Vdash \bullet Y\rangle}{\updownarrow \frac{\langle \mathcal{R}^*(\Pi) \fatsemi \wr X \fatsemi \mathcal{R} \mid \overline{\mathbb{N}}_1 \int \Vdash \circ Y\rangle}{\updownarrow \frac{\langle X(\mathcal{R}^*(\Pi)) \fatsemi \star\varepsilon \Vdash Y\rangle}{[\mathcal{R} \setminus Y](X(\mathcal{R}^*(\Pi))) = \varnothing}} \wedge X(\mathcal{R}^*(\Pi)) \subseteq \mathcal{R}^{-1}(\mathcal{T})\,.}}}$$

Optional global **optimisation** phase: $\mathcal{R}^{-1}(\mathcal{T}) \to Y^{-1}(\mathcal{T})$.

Translatable Fragment

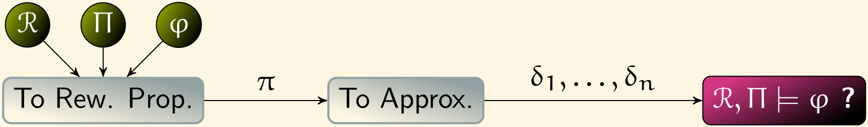
Exactly rewrite-translatable fragment:

$$X \in \wp(\mathcal{R}), \;\; m \in \mathbb{N}$$
$$\varphi := \top \mid \bot \mid X \mid \neg X \mid \varphi \wedge \varphi \mid \psi \Rightarrow \varphi \mid \bullet\varphi \mid \circ\varphi \mid \Box\,\varphi$$
$$\psi := \top \mid \bot \mid X \mid \neg X \mid \psi \vee \psi \mid \psi \wedge \psi \mid \bullet\psi \mid \circ\psi \mid \Phi$$
$$\Phi := \text{at least } \varepsilon\text{-stabilisable } \Box\,\varphi$$

Practical pre-experimental evaluation:
good partial support of [Dwyer et al., 1999] patterns.

LTL on Rewrite Sequences

Perspectives (Translation Into Rewrite Proposition)

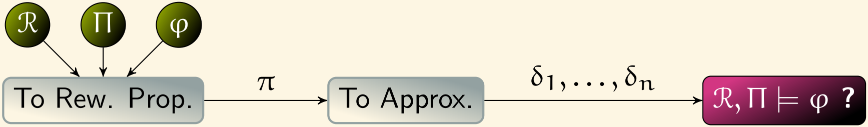


- [Héam et al., 2012a] Int. Conf. IJCAR'12, Manchester

- **Extensions**: Past-Time and Existential LTL
- Dealing with **eventuality** by studying "exhaustion": e.g. $\Diamond \neg\{f(x) \to x\}$ holds with bounded f-height & no intro

LTL on Rewrite Sequences

Perspectives (Approximated Decision Procedures)



- Coping with more **non-linearity** – e.g. protocols, rewrite steps e.g. $f(x, x) \to g(x)$, $f(x) \to g(x, x)$,. . .
- Tractable algorithmic toolbox for **TAGE**

Last points ⇒ closer study of **TAGE complexity**

1. Model-Checking LTL on Rewrite Sequences
   - Statement of the Central Problem
   - Our Approach: An Overview

2. TAGE With a Bounded Number of Constraints
   - Global Equality Constraints
   - Overview of the Results

3. Other Works and Some Perspectives
   - Results on SAT & Tree-Walking Automata
   - Perspectives and Questions

Tree Automata

With Global Equality Constraints [Filiot et al., 2008]

TAGE, TA$^=$, Positive TAGED, $\mathcal{A} = \langle \mathbb{A}, \mathbf{Q}, \mathrm{F}, \boldsymbol{\Delta}, \approxeq \rangle$:

$\langle \mathbb{A}, \mathbf{Q}, \mathrm{F}, \boldsymbol{\Delta} \rangle$	vanilla **tree automaton** $\mathrm{ta}(\mathcal{A})$
$\approxeq$	equality **constraints**, $\approxeq \subseteq \mathrm{Q}^2$

Constraint $\mathbf{p} \approxeq \mathbf{q}$:

run ρ of $\mathcal{A}$ on t:

- **run** of $\mathrm{ta}(\mathcal{A})$ on t
- **satisfying** $\approxeq$: $\forall \alpha, \beta \in \mathcal{P}(t);\ \rho(\alpha) \approxeq \rho(\beta) \Rightarrow t|_\alpha = t|_\beta$

accepting run: accepting for $\mathrm{ta}(\mathcal{A})$

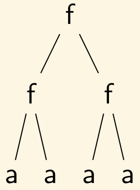
Tree Automata

With Global Equality Constraints [Filiot et al., 2008]

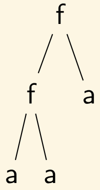
$\mathbb{A} = \{\, a/_0, f/_2 \,\}$, $Q = \{\, q, \hat{q}, q_f \,\}$, $F = \{\, q_f \,\}$, $\hat{q} \approxeq \hat{q}$, and

$$\Delta = \{\, f(\hat{q}, \hat{q}) \to q_f,\ f(q, q) \to q,\ f(q, q) \to \hat{q},\ a \to q,\ a \to \hat{q} \,\}$$

$u =$



and $v =$



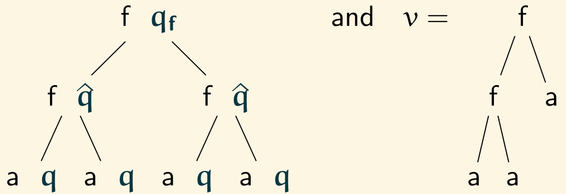
Tree Automata

With Global Equality Constraints [Filiot et al., 2008]

$\mathbb{A} = \{ a/_0, f/_2 \}$, $Q = \{ q, \hat{q}, q_f \}$, $F = \{ q_f \}$, $\hat{q} \approxeq \hat{q}$, and

$$\Delta = \{ f(\hat{q}, \hat{q}) \to q_f,\ f(q, q) \to q,\ f(q, q) \to \hat{q},\ a \to q,\ a \to \hat{q} \}$$

$u, \rho_u =$ and $v =$



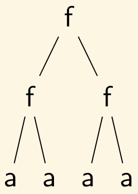
Tree Automata

With Global Equality Constraints [Filiot et al., 2008]

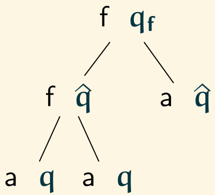
$\mathbb{A} = \{\, a/_0, f/_2 \,\}$, $Q = \{\, q, \hat{q}, q_f \,\}$, $F = \{\, q_f \,\}$, $\hat{q} \approx \hat{q}$, and

$$\Delta = \{\, f(\hat{q}, \hat{q}) \to q_f,\ f(q,q) \to q,\ f(q,q) \to \hat{q},\ a \to q,\ a \to \hat{q} \,\}$$

$u =$



and $v, \rho_v =$



1. Model-Checking LTL on Rewrite Sequences
   - Statement of the Central Problem
   - Our Approach: An Overview

2. TAGE With a Bounded Number of Constraints
   - Global Equality Constraints
   - Overview of the Results

3. Other Works and Some Perspectives
   - Results on SAT & Tree-Walking Automata
   - Perspectives and Questions

$TA^=$ versus $TA^=_k$

Restriction on the **kind** of constraints: Rigid Automata (RTA)

- Same expressive power as $TA^=$
- Less compact representations
- Linear emptiness / finiteness tests, vs. ExpTime-complete
- Applications: [Jacquemard et al., 2009, Vacher, 2010]

What of the **number** of constraints? $TA^=_k$ $\mathcal{A} = \langle \Sigma, Q, F, \Delta, \approxeq \rangle$:

$\langle \Sigma, Q, F, \Delta, \approxeq \rangle$	$TA^=$ $\mathcal{A}$
$\approxeq$	such that $\mathrm{Card}(\approxeq) \leqslant k$

Summary of Results

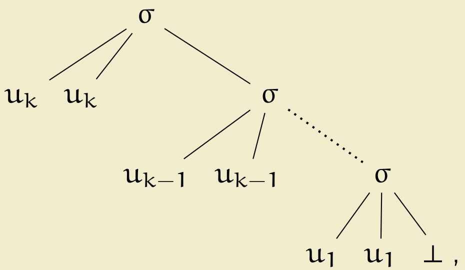
- [Héam et al., 2012c] Int. Conf. CIAA'12, Porto

- **Strict hierarchy** of powers: $\mathcal{L}(\mathsf{TA}^{\equiv}_{k}) \subset \mathcal{L}(\mathsf{TA}^{\equiv}_{k+1})$
- **Emptiness** linear for $\mathsf{TA}^{\equiv}_{1}$, ExpTime-complete $\mathsf{TA}^{\equiv}_{2}$
- **Finiteness** polynomial for $\mathsf{TA}^{\equiv}_{1}$, ExpTime-complete for $\mathsf{TA}^{\equiv}_{2}$
- NP-complete **membership** becomes polynomial if k fixed.

Summary of Results

- [Héam et al., 2012c] Int. Conf. CIAA'12, Porto

- **Strict hierarchy** of powers: $\mathcal{L}(\mathsf{TA}^{\equiv}_{k}) \subset \mathcal{L}(\mathsf{TA}^{\equiv}_{k+1})$



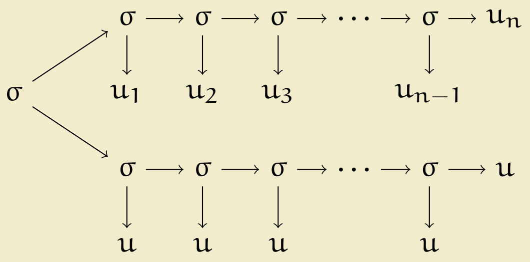
$u_i \in \mathcal{T}(\mathbb{A}_i)\ \forall i$

- **Emptiness** linear for $\mathsf{TA}^{\equiv}_{1}$, ExpTime-complete $\mathsf{TA}^{\equiv}_{2}$
- **Finiteness** polynomial for $\mathsf{TA}^{\equiv}_{1}$, ExpTime-complete for $\mathsf{TA}^{\equiv}_{2}$
- NP-complete **membership** becomes polynomial if k fixed.

Summary of Results

- [Héam et al., 2012c] Int. Conf. CIAA'12, Porto

- **Strict hierarchy** of powers: $\mathcal{L}(\mathrm{TA}_k^{\equiv}) \subset \mathcal{L}(\mathrm{TA}_{k+1}^{\equiv})$
- **Emptiness** linear for $\mathrm{TA}_1^{\equiv}$, ExpTime-complete $\mathrm{TA}_2^{\equiv}$



- **Finiteness** polynomial for $\mathrm{TA}_1^{\equiv}$, ExpTime-complete for $\mathrm{TA}_2^{\equiv}$
- NP-complete **membership** becomes polynomial if k fixed.

Summary of Results

- [Héam et al., 2012c] Int. Conf. CIAA'12, Porto

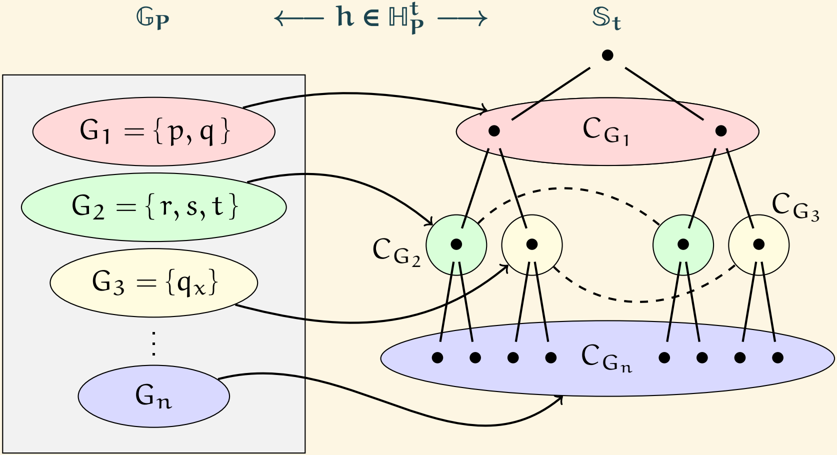
- **Strict hierarchy** of powers: $\mathcal{L}(\mathrm{TA}_k^{\equiv}) \subset \mathcal{L}(\mathrm{TA}_{k+1}^{\equiv})$
- **Emptiness** linear for $\mathrm{TA}_1^{\equiv}$, ExpTime-complete $\mathrm{TA}_2^{\equiv}$
- **Finiteness** polynomial for $\mathrm{TA}_1^{\equiv}$, ExpTime-complete for $\mathrm{TA}_2^{\equiv}$

Reduction of emptiness to finiteness.

- NP-complete **membership** becomes polynomial if k fixed.

Summary of Results

- NP-complete **membership** becomes polynomial if k fixed.



1. Model-Checking LTL on Rewrite Sequences
   - Statement of the Central Problem
   - Our Approach: An Overview

2. TAGE With a Bounded Number of Constraints
   - Global Equality Constraints
   - Overview of the Results

3. Other Works and Some Perspectives
   - Results on SAT & Tree-Walking Automata
   - Perspectives and Questions

TAGE SAT & Tree-Walking Overloops

- [Héam et al., 2010] Int. Workshop CSTVA'10, Paris
- [Héam et al., 2011] Int. Conf. CIAA'11, Blois
- [Héam et al., 2012b] Int. Journal Theo. Comp. Sci.

- **SAT Encoding** for TAGE membership & optimisations.
- Formal treatment of tree-walking **loops** for transformation into bottom-up TA; revealed missing factor in space $\Sigma \times \mathbb{T} \times 2^{Q^2}$.
- Introduced tree-walking **overloops**: restores $\mathbb{T} \times 2^{Q^2}$, smaller automata in practice in extensive random tests.
- Shown overloops **upper-bound** is $|\mathbb{T}| \cdot 2^{|Q| \log_2(|Q|+1)}$ in the deterministic case. Note that exponential is unavoidable.
- Polynomial overloops-based **approximation** to TWA emptiness, vs. ExpTime-c. Very precise in random tests.

Polynomial Approximation for Emptiness

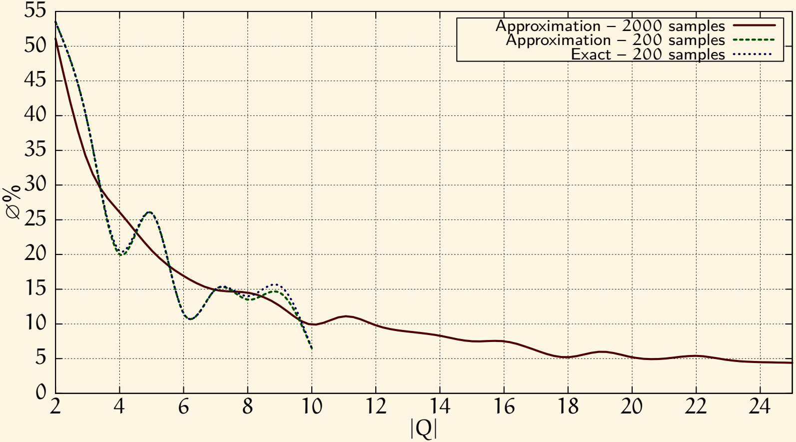
Random tests

1. **Ad-hoc** scheme: $\approx 20\,000$ TWA, $2 \leqslant |Q| \leqslant 20$, $|\Delta| \approx 3 \times |Q|$, 75% of empty languages, only two *Unknown* instead of *Empty*.

2. **Uniform** scheme [Héam et al., 2009], REGAL back-end for FSA generation [Bassino et al., 2007]. $2\,000$ deterministic and complete TWA uniformly generated for each $2 \leqslant |Q| \leqslant 25$.

Polynomial Approximation for Emptiness

Random tests



Size Comparison: Loops vs. Overloops

One Example & Uniform Generation Scheme

For $\mathcal{X}$: loops $\|\mathcal{B}_\mathsf{l}\| = 1986$; overloops $\|\mathcal{B}_\mathsf{o}\| = 95$; deterministic minimal $\|\mathcal{B}_\mathsf{m}\| = 56$; smallest known non-deterministic $\|\mathcal{B}_\mathsf{s}\| = 34$.

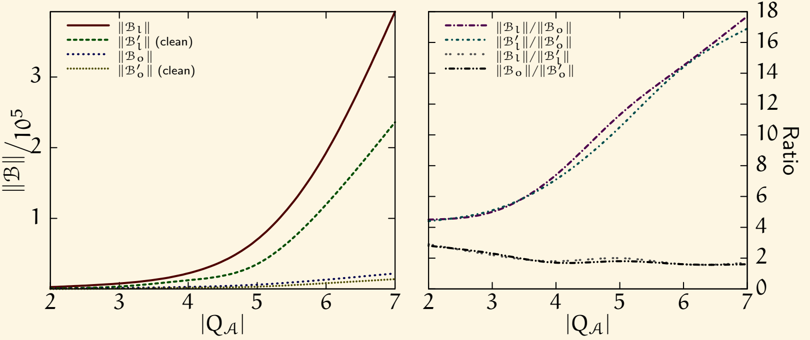
Loops **60 times** worse than manual optimal; overloops **3 times**.

Orthogonal to **post-processing** cleanup: $\|\mathcal{B}_\mathsf{l}'\| = 1617$, $\|\mathcal{B}_\mathsf{o}'\| = 78$.

$$\frac{\|\mathcal{B}_\mathsf{l}\|}{\|\mathcal{B}_\mathsf{o}\|} \approx 20.9 \quad \text{and} \quad \frac{\|\mathcal{B}_\mathsf{l}'\|}{\|\mathcal{B}_\mathsf{o}'\|} \approx 20.7 \quad \text{and} \quad \frac{\|\mathcal{B}_\mathsf{l}\|}{\|\mathcal{B}_\mathsf{l}'\|} \approx \frac{\|\mathcal{B}_\mathsf{o}\|}{\|\mathcal{B}_\mathsf{o}'\|} \approx 1.2 \,.$$

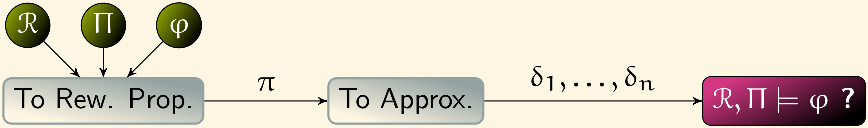
Size Comparison: Loops vs. Overloops

One Example & Uniform Generation Scheme



Conclusion / Summary

Generalisation of the translation



Study of complexity of **bounded global constraints**

Improved loops-based methods for **tree-walking automata**

Perspectives

Full TAGE may not be required for $X(\Pi)$; **flat constraints** ensure polynomial emptiness decision; are they enough?

Implemented algorithmic **toolbox** for these automata.

Rewrite propositions go beyond LTL (e.g. ∃-LTL).
What is their **full expressive power**?

Intermix **state** and **transition**-based properties.

Questions ?

Supported Fragment, In Practice

Partially Supported Patterns From [Dwyer et al., 1999]

	Scope					
Pattern	Global	Before	After	Between	Until	*Support*
Absence	**41**	5	12	18	9	48%
Universality	**110**	1	**5**	2	1	96%
Existence	12	1	4	8	1	0%
Bound Exist.	0	0	0	1	0	0%
Response	**241**	1	**3**	0	0	99%
Precedence	**25**	0	1	0	0	96%
Resp. Chain	8	0	0	0	0	0%
Prec. Chain	1	0	0	0	0	0%
Support	95%	0%	32%	0%	0%	83%

Formal Tools for Verification

Reliable Software

Software **failure** is **undesirable**. . .

Ariane 5, Therac-25, Mariner I, Phobos I, XA/21 USA & Canada Northeast 2003 blackout, MIM-104 Patriot anti-missile, Mars Climate Orbiter, Mars Polar Lander, Mars Global Surveyor space probes,. . .

. . . hence the need for **formal verification** methods.

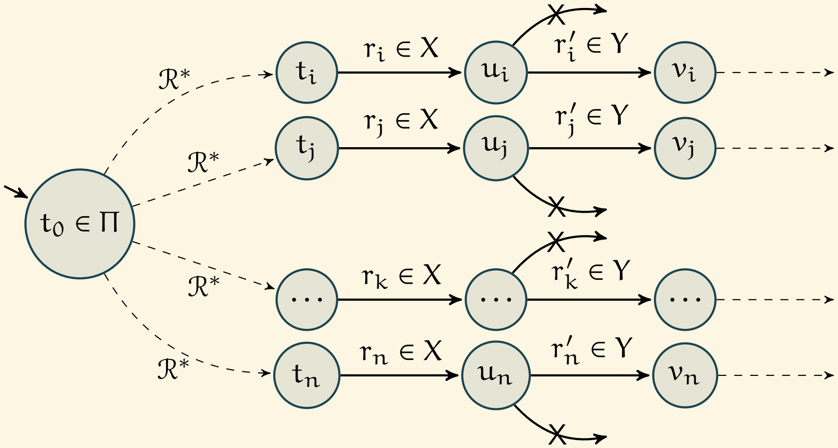
E.G. With **Hoare logic**, correctness is a mathematical theorem.

Precondition, code, post-condition: $\{\top\}\, x := y\, \{x = y\}$.

Manual proofs require mathematical ingenuity. **Automation**?

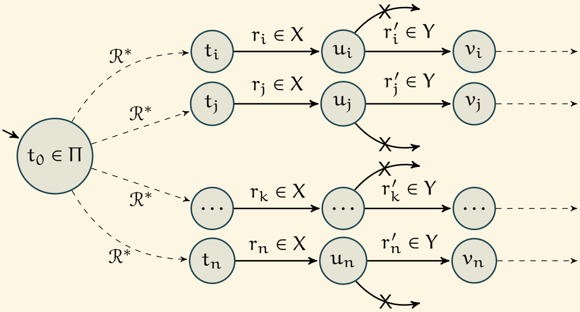
Model-Checking Rewrite Sequences

Coding the Behaviour of the System: $\square(X \Rightarrow \bullet Y)$



Maximal Rewrite Words

Coding the Behaviour of the System



Executions may or may not **terminate**: finite and infinite words.

Maximal Rewrite Words

Coding the Behaviour of the System

Finite or infinite **words** on $\mathcal{R}$:

$$\overline{\mathbb{N}} = \mathbb{N} \cup \{+\infty\} \qquad \mathcal{W} = \bigcup_{n \in \overline{\mathbb{N}}} (\llbracket 1, n \rrbracket \to \mathcal{R})$$

Notation: **length** $\#w \in \overline{\mathbb{N}}$: $\#w = \mathrm{Card}(\mathrm{dom}\, w)$.

Maximal rewrite words of $\mathcal{R}$, originating in Π:

$(\!|\Pi|\!)$ is the set of words $w \in \mathcal{W}$ such that

$$\exists u_0 \in \Pi : \exists u_1, \dots, u_{\#w} \in \mathcal{T} : \forall k \in \mathrm{dom}\, w,$$
$$u_{k-1} \xrightarrow{w(k)} u_k \;\wedge\; \#w \in \mathbb{N} \Rightarrow \mathcal{R}(\{u_{\#w}\}) = \varnothing$$

Syntax and Semantics for LTL

Close to Finite-LTL [Manna and Pnueli, 1995]

$$\varphi := X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \bullet^m \varphi \mid \circ^m \varphi \mid \varphi \,\mathbf{U}\, \varphi \qquad X \in \wp(\mathcal{R})$$

$$\top \mid \bot \mid \varphi \vee \varphi \mid \varphi \Rightarrow \varphi \mid \Diamond\,\varphi \mid \Box\,\varphi \qquad m \in \mathbb{N}\,.$$

$$\begin{aligned}
(w,i) &\models X && \Leftrightarrow && i \in \operatorname{dom} w \text{ and } w(i) \in X \\
(w,i) &\models \neg\varphi && \Leftrightarrow && (w,i) \not\models \varphi \\
(w,i) &\models (\varphi \wedge \psi) && \Leftrightarrow && (w,i) \models \varphi \text{ and } (w,i) \models \psi \\
(w,i) &\models \bullet^m \varphi && \Leftrightarrow && i+m \in \operatorname{dom} w \text{ and } (w,i+m) \models \varphi \\
(w,i) &\models \circ^m \varphi && \Leftrightarrow && i+m \notin \operatorname{dom} w \text{ or } (w,i+m) \models \varphi \\
(w,i) &\models \varphi \,\mathbf{U}\, \psi && \Leftrightarrow && \begin{cases} \exists j \in \operatorname{dom} w : j \geqslant i \wedge (w,j) \models \psi \\ \quad \wedge\ \forall k \in [\![i, j-1]\!],\ (w,k) \models \varphi \end{cases}
\end{aligned}$$

For any $w \in \mathcal{W}$, $i \in \mathbb{N}_1$, $m \in \mathbb{N}$ and $X \in \wp(\mathcal{R})$.

Syntax and Semantics for LTL

Close to Finite-LTL [Manna and Pnueli, 1995]

$$\varphi := X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \bullet^m \varphi \mid \circ^m \varphi \mid \varphi \,\mathbf{U}\, \varphi \qquad X \in \wp(\mathcal{R})$$
$$\top \mid \bot \mid \varphi \vee \varphi \mid \varphi \Rightarrow \varphi \mid \Diamond\, \varphi \mid \Box\, \varphi \qquad m \in \mathbb{N}\,.$$

$$
\begin{array}{lcl}
(w,i) \models X & \Leftrightarrow & i \in \operatorname{dom} w \text{ and } w(i) \in X \\
(w,i) \models \neg\varphi & \Leftrightarrow & (w,i) \not\models \varphi \\
(w,i) \models (\varphi \wedge \psi) & \Leftrightarrow & (w,i) \models \varphi \text{ and } (w,i) \models \psi \\
(w,i) \models \bullet^m \varphi & \Leftrightarrow & i+m \in \operatorname{dom} w \text{ and } (w,i+m) \models \varphi \\
(w,i) \models \circ^m \varphi & \Leftrightarrow & i+m \notin \operatorname{dom} w \text{ or } (w,i+m) \models \varphi \\
(w,i) \models \Box\, \varphi & \Leftrightarrow & \forall j \in \operatorname{dom} w,\ j \geqslant i \Rightarrow (w,j) \models \varphi
\end{array}
$$

For any $w \in \mathcal{W}$, $i \in \mathbb{N}_1$, $m \in \mathbb{N}$ and $X \in \wp(\mathcal{R})$.

Syntax and Semantics for LTL

Close to Finite-LTL [Manna and Pnueli, 1995]

$$
\begin{array}{lcl}
(w, i) \models X & \Leftrightarrow & i \in \operatorname{dom} w \text{ and } w(i) \in X \\
(w, i) \models \neg\varphi & \Leftrightarrow & (w, i) \not\models \varphi \\
(w, i) \models (\varphi \wedge \psi) & \Leftrightarrow & (w, i) \models \varphi \text{ and } (w, i) \models \psi \\
(w, i) \models \bullet^m \varphi & \Leftrightarrow & i + m \in \operatorname{dom} w \text{ and } (w, i + m) \models \varphi \\
(w, i) \models \circ^m \varphi & \Leftrightarrow & i + m \notin \operatorname{dom} w \text{ or } (w, i + m) \models \varphi \\
(w, i) \models \square\, \varphi & \Leftrightarrow & \forall j \in \operatorname{dom} w,\ j \geqslant i \Rightarrow (w, j) \models \varphi
\end{array}
$$

For any $w \in \mathcal{W}$, $i \in \mathbb{N}_1$, $m \in \mathbb{N}$ and $X \in \wp(\mathcal{R})$.

Satisfaction:

- $w \models \varphi \iff (w, 1) \models \varphi$
- $\mathcal{R}, \Pi \models \varphi \iff \forall w \in (\!|\Pi|\!),\ w \models \varphi$

Rewrite Propositions

Problem Statement: First Translation Step

Rewrite proposition π on $\mathcal{R}$, from Π; has a trivial truth value

$$\pi := \gamma \mid \gamma \wedge \gamma \mid \gamma \vee \gamma \qquad \gamma := \ell = \varnothing \mid \ell \subseteq \ell$$
$$X \in \wp(\mathcal{R}) \qquad \ell := \Pi \mid \mathcal{T} \mid X(\ell) \mid X^{-1}(\ell) \mid X^{*}(\ell)$$

Problem statement: translations into RP

Input: $\mathcal{R}$, $\varphi \in$ LTL, $\Pi \subseteq \mathcal{T}$ **Output:** RP π such that:

$\mathcal{R}, \Pi \models \varphi \iff \pi$ (exact translation)
$\mathcal{R}, \Pi \models \varphi \Longleftarrow \pi$ (under-approximated translation)
$\mathcal{R}, \Pi \models \varphi \implies \pi$ (*over-approximated translation*)

Intuitions for the Translation

Boundaries of the Translatable Fragment

$\mathcal{R}^*(\Pi)$ hides **traces**:
$\Diamond X$ probably untranslatable. So are $\{ \Diamond, \mathbf{U}, \mathbf{W}, \mathbf{R}, \ldots \}$.

Formulæ in **sanitised form**: negation on literals. **Not** exactly NNF.

$$(A \vee B) \Rightarrow C \qquad (A \Rightarrow C) \wedge (B \Rightarrow C) \qquad (\neg A \wedge \neg B) \vee C$$

Preprocessing to fit translatable **"antecedent/consequent"** form.

Signatures

Implication: Girdling the Future

Idea: $\varphi \Rightarrow \psi$? φ as an *assumption*, i.e. a *model* of φ: $\xi(\varphi)$

$$\Sigma = \bigcup_{n \in \mathbb{N}} \Big[\big([\![1, n]\!] \cup \{\omega\} \big) \to \wp(\mathcal{R}) \Big] \times \wp(\overline{\mathbb{N}}) \ .$$

Notations: $\sigma \in \Sigma$

- compactly as $\sigma = \wr f \mid S \int = \wr \partial\sigma \mid \nabla\sigma \int$,
- or *in extenso* as $\wr f(1), f(2), \ldots, f(\#\sigma) \,⨾\, f(\omega) \mid S \int$.

Example: $\xi\big(X \wedge \circ^1 Y \wedge \circ^2 \square Z\big) = \wr X, Y \,⨾\, Z \mid \overline{\mathbb{N}}_1 \int$

Signatures

Implication: Girdling the Future

$$\Sigma = \bigcup_{n \in \mathbb{N}} \Big[\big([\![1, n]\!] \cup \{\omega\} \big) \to \wp(\mathcal{R}) \Big] \times \wp(\overline{\mathbb{N}}) \,.$$

Notations: $\sigma \in \Sigma$

- compactly as $\sigma = \wr f \mid S \int = \wr \partial\sigma \mid \nabla\sigma \int$,
- or *in extenso* as $\wr f(1), f(2), \dots, f(\#\sigma) \,⨾\, f(\omega) \mid S \int$.

Example: $\xi\big(X \wedge \circ^1 Y \wedge \circ^2 \Box Z\big) = \wr X, Y \,⨾\, Z \mid \overline{\mathbb{N}}_1 \int$

Constrained Words:

$(\!| \Pi \,⨾\, \sigma |\!) = \{ w \in (\!| \Pi |\!) \mid \#w \in \nabla\sigma \;\wedge\; \forall k \in \operatorname{dom} w,\; w(k) \in \sigma[k] \}$

$\forall\, \Pi \subseteq \mathcal{T}, \varphi \in \mathcal{A}\text{-LTL},\; (\!| \Pi \,⨾\, \xi(\varphi) |\!) = \{ w \in (\!| \Pi |\!) \mid w \models \varphi \}$

Signatures: the Transformation $\xi(\cdot)$

Modelling the Antecedent to Girdle the Future

$$\xi(\top) = \wr \fatsemi \mathcal{R} \mid \overline{\mathbb{N}} \int = \varepsilon \qquad \xi(\bot) = \wr \fatsemi \varnothing \mid \varnothing \int$$

$$\xi(X) = \wr X \fatsemi \mathcal{R} \mid \overline{\mathbb{N}}_1 \int \qquad \xi(\neg X) = \wr \mathcal{R} \setminus X \fatsemi \mathcal{R} \mid \overline{\mathbb{N}} \int$$

$$\xi(\bullet^m \varphi) = \xi(\varphi) \blacktriangleright m \qquad \xi(\circ^m \varphi) = \xi(\varphi) \triangleright m$$

$$\xi(\varphi \wedge \psi) = \xi(\varphi) \otimes \xi(\psi) \qquad \xi(\square\, \varphi) = \bigotimes_{m=0}^{\infty} \Big[\xi(\varphi) \triangleright m \Big]$$

Signatures: the Transformation $\xi(\cdot)$

Modelling the Antecedent to Girdle the Future

$$\xi(\top) = \wr\,{}_{⨾}\mathcal{R} \mid \overline{\mathbb{N}}\,\int = \varepsilon \qquad \xi(\bot) = \wr\,{}_{⨾}\varnothing \mid \varnothing\,\int$$

$$\xi(X) = \wr X \,⨾\, \mathcal{R} \mid \overline{\mathbb{N}}_1\,\int \qquad \xi(\neg X) = \wr \mathcal{R} \setminus X \,⨾\, \mathcal{R} \mid \overline{\mathbb{N}}\,\int$$

$$\xi(\bullet^m \varphi) = \xi(\varphi) \blacktriangleright m \qquad \xi(\circ^m \varphi) = \xi(\varphi) \rhd m$$

$$\xi(\varphi \wedge \psi) = \xi(\varphi) \otimes \xi(\psi) \qquad \xi(\Box\, \varphi) = \bigotimes_{m=0}^{\infty} \left[\xi(\varphi) \rhd m\right]$$

- **$\sigma \blacktriangleright m$ = Strong Shift Right =** $\wr \mathcal{R}_1, \dots, \mathcal{R}_m, \partial\sigma(1), \dots, \partial\sigma(\#\sigma) \,⨾\, \partial\sigma(\omega) \mid (\nabla\sigma \setminus \{0\}) + m\,\int$
- **$\sigma \rhd m$ = Weak Shift Right =** $\wr \mathcal{R}_1, \dots, \mathcal{R}_m, \partial\sigma(1), \dots, \partial\sigma(\#\sigma) \,⨾\, \partial\sigma(\omega) \mid [\![0, m]\!] \cup (\nabla\sigma + m)\,\int$

Signatures: the Transformation $\xi(\cdot)$

Modelling the Antecedent to Girdle the Future

$$\xi(\top) = \wr\, ⨾ \mathcal{R} \mid \overline{\mathbb{N}} \int = \varepsilon \qquad \xi(\bot) = \wr\, ⨾ \varnothing \mid \varnothing \int$$

$$\xi(X) = \wr X ⨾ \mathcal{R} \mid \overline{\mathbb{N}}_1 \int \qquad \xi(\neg X) = \wr \mathcal{R} \setminus X ⨾ \mathcal{R} \mid \overline{\mathbb{N}} \int$$

$$\xi(\bullet^m \varphi) = \xi(\varphi) \blacktriangleright m \qquad \xi(\circ^m \varphi) = \xi(\varphi) \triangleright m$$

$$\xi(\varphi \wedge \psi) = \xi(\varphi) \otimes \xi(\psi) \qquad \xi(\square\, \varphi) = \bigotimes_{m=0}^{\infty} \left[\xi(\varphi) \triangleright m \right]$$

Product Property: $(\!|\Pi ⨾ \sigma \otimes \sigma'|\!) = (\!|\Pi ⨾ \sigma|\!) \cap (\!|\Pi ⨾ \sigma'|\!)$

Example: $\sigma = \wr X, Y ⨾ Z \mid \mathbb{N}_2 \int \qquad \rho = \wr X' ⨾ Z' \mid \mathbb{N}_3 \int$

$$\sigma \otimes \rho = \wr X \cap X', Y \cap Z' ⨾ Z \cap Z' \mid \mathbb{N}_3 \int$$

Signatures: the Transformation $\xi(\cdot)$

Modelling the Antecedent to Girdle the Future

$$\xi(\top) = \wr\, ⨾\mathcal{R} \mid \overline{\mathbb{N}} \,\wr = \varepsilon \qquad \xi(\bot) = \wr\, ⨾\varnothing \mid \varnothing \,\wr$$

$$\xi(X) = \wr\, X ⨾ \mathcal{R} \mid \overline{\mathbb{N}}_1 \,\wr \qquad \xi(\neg X) = \wr\, \mathcal{R} \setminus X ⨾ \mathcal{R} \mid \overline{\mathbb{N}} \,\wr$$

$$\xi(\bullet^m \varphi) = \xi(\varphi) \blacktriangleright m \qquad \xi(\circ^m \varphi) = \xi(\varphi) \triangleright m$$

$$\xi(\varphi \wedge \psi) = \xi(\varphi) \otimes \xi(\psi) \qquad \xi(\square\, \varphi) = \bigotimes_{m=0}^{\infty} \Big[\xi(\varphi) \triangleright m\Big]$$

$$\square\, \varphi \Leftrightarrow \bigwedge_{m=0}^{\infty} \circ^m \varphi \qquad (\!|\Pi ⨾ \bigotimes_{n=0}^{\infty} \sigma_n|\!) = \bigcap_{n=0}^{\infty} (\!|\Pi ⨾ \sigma_n|\!)$$

$$\bigotimes_{n=0}^{\infty} [\sigma \blacktriangleright n] \quad \text{and} \quad \bigotimes_{n=0}^{\infty} [\sigma \triangleright n] \quad \text{converge } \forall \sigma \in \Sigma$$

Rewrite Proposition → Procedure

automatic kind inference and generation rules

Kind inference: expressiveness required & assumptions

$$\alpha : \mathsf{TA} \vdash X(\alpha) : \mathsf{TA}^{=} \lhd \qquad \alpha : \mathsf{TA}, X : \text{reg-pres} \vdash X(\alpha) : \mathsf{TA}$$

$$\vdash X^{-1}(\mathcal{T}) : \mathsf{TA}^{=} \lhd \qquad X : \text{left-lin} \vdash X^{-1}(\mathcal{T}) : \mathsf{TA}$$

$$\alpha : \mathsf{TA} \vdash \natural\alpha : \mathsf{TA} \qquad \alpha : \mathsf{TA}^{=} \vdash \natural\alpha : \mathsf{TA}, \natural\alpha : +$$

Procedure Generation: from languages to automata

$$\Gamma \fatsemi X^{-1}(\mathcal{T}) \rightrightarrows \Gamma, \langle X : \text{left-lin} \rangle \fatsemi X^{-1}(\mathcal{T})$$

$$\Gamma \fatsemi [\ell \rightarrowtail \Delta, \alpha] \fatsemi \Delta \vdash^{*} \alpha : \mathsf{TA} \fatsemi X(\ell) \rightrightarrows \Gamma, \Delta, \langle X : \text{reg-pres} \rangle \fatsemi X(\alpha)$$

$$\Gamma \fatsemi [\ell \rightarrowtail \Delta, \alpha] \fatsemi \Delta \vdash^{*} \alpha : \mathsf{TA}^{=} \fatsemi X(\ell) \rightrightarrows \Gamma, \Delta, \langle X : \text{reg-pres} \rangle \fatsemi X(\natural\alpha)$$

Supported Fragment, In Practice

Partially Supported Patterns From [Dwyer et al., 1999]

	Scope					
Pattern	Global	Before	After	Between	Until	*Support*
Absence	**41**	5	12	18	9	48%
Universality	**110**	1	**5**	2	1	96%
Existence	12	1	4	8	1	0%
Bound Exist.	0	0	0	1	0	0%
Response	**241**	1	**3**	0	0	99%
Precedence	**25**	0	1	0	0	96%
Resp. Chain	8	0	0	0	0	0%
Prec. Chain	1	0	0	0	0	0%
Support	95%	0%	32%	0%	0%	83%

Tree Automata

[Comon et al., 2008]

Introduced in the fifties; **regular tree languages**:

- model-checking: programs, protocols,...
- automated theorem-proving
- XML schema and (esp. variants) query languages
- ...and so much more

Doesn't deal with **comparisons** and **non-linearity**:

- $\{ f(u,u) \mid u \in \mathcal{T}(\Sigma) \}$ e.g. password verification
- $\{ f(u,v) \mid u,v \in \mathcal{T}(\Sigma), u \neq v \}$ e.g. primary keys
- $\mathcal{R}(\ell)$, ℓ regular, $\mathcal{R}$ a TRS e.g. $\{ g(x) \to f(x,x) \}(\mathcal{T}(\mathbb{A}))$

Tree Automata

Bottom-Up, Non-Deterministic, Finite

Tree Automaton $\mathcal{A} = \langle \mathbb{A}, \mathbf{Q}, \mathbf{F}, \boldsymbol{\Delta} \rangle$:

$\mathbb{A}$	finite **ranked alphabet**
$\mathbf{Q}$	finite set of **states**
$\mathbf{F}$	**final** states, $F \subseteq Q$
$\boldsymbol{\Delta}$	finite set of **transitions**

Transition $\mathbf{r} \in \boldsymbol{\Delta}$:

$$\sigma(q_1, \dots, q_n) \to q \qquad \sigma \in \mathbb{A}_n \quad q_1, \dots, q_n, q \in Q$$

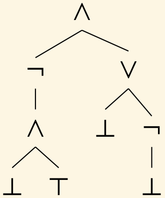
Tree Automata

Bottom-Up, Non-Deterministic, Finite

$\mathbb{A} = \{ \wedge, \vee/2, \neg/1, \top, \bot/0 \}$, $Q = \{ q_0, q_1 \}$, $F = \{q_1\}$, $\Delta =$

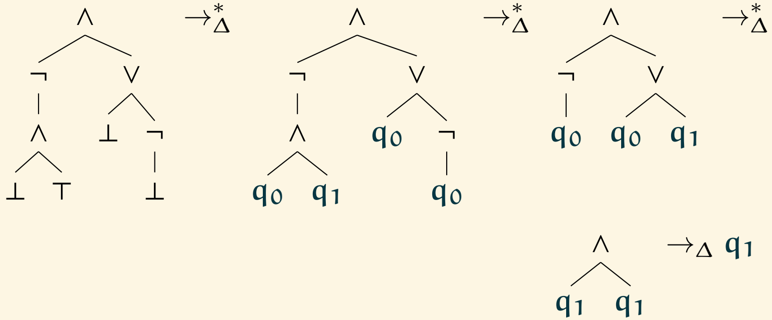
$$\left\{ \begin{array}{l} \top \to q_1, \quad \bot \to q_0, \quad \neg(q_b) \to q_{\neg b} \\ \wedge(q_b, q_{b'}) \to q_{b \wedge b'}, \quad \vee(q_b, q_{b'}) \to q_{b \vee b'} \end{array} \;\middle|\; b, b' \in \{0, 1\} \right\}$$

$t =$



Tree Automata

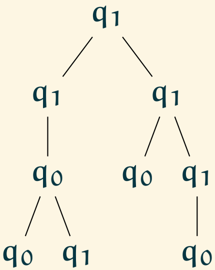
Bottom-Up, Non-Deterministic, Finite



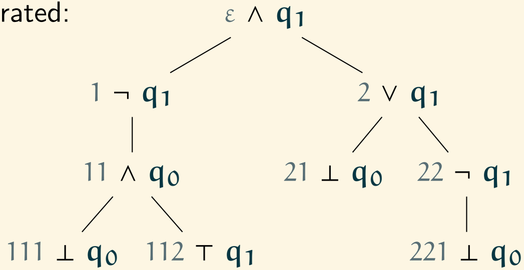
Tree Automata

Runs and Languages

The reduction $t \to_{\Delta}^{*} q_1$ is captured by the **run**:



decorated:



Tree Automata

With Global Equality Constraints [Filiot et al., 2008]

TAGE, TA$^=$, Positive TAGED, $\mathcal{A} = \langle \mathbb{A}, Q, F, \Delta, \approxeq \rangle$:

$\langle \mathbb{A}, Q, F, \Delta \rangle$	vanilla **tree automaton** $\text{ta}(\mathcal{A})$
$\approxeq$	equality **constraints**, $\approxeq \subseteq Q^2$

Constraint $p \approxeq q$:

run ρ of $\mathcal{A}$ on t:

- **run** of $\text{ta}(\mathcal{A})$ on t
- **satisfying** $\approxeq$: $\forall \alpha, \beta \in \mathcal{P}(t);\ \rho(\alpha) \approxeq \rho(\beta) \Rightarrow t|_\alpha = t|_\beta$

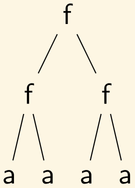
accepting run: accepting for $\text{ta}(\mathcal{A})$

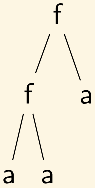
Tree Automata

With Global Equality Constraints [Filiot et al., 2008]

$\mathbb{A} = \{ a/_0, f/_2 \}$, $Q = \{ q, \hat{q}, q_f \}$, $F = \{ q_f \}$, $\hat{q} \approxeq \hat{q}$, and

$$\Delta = \{ f(\hat{q}, \hat{q}) \to q_f,\ f(q, q) \to q,\ f(q, q) \to \hat{q},\ a \to q,\ a \to \hat{q} \}$$

$u =$ 

and $v =$ 

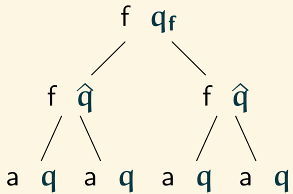
Tree Automata

With Global Equality Constraints [Filiot et al., 2008]

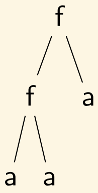
$\mathbb{A} = \{ a/_0, f/_2 \}$, $Q = \{ q, \hat{q}, q_f \}$, $F = \{ q_f \}$, $\hat{q} \approxeq \hat{q}$, and

$$\Delta = \{ f(\hat{q}, \hat{q}) \to q_f,\ f(q,q) \to q,\ f(q,q) \to \hat{q},\ a \to q,\ a \to \hat{q} \}$$

$u, \rho_u =$



and $v =$

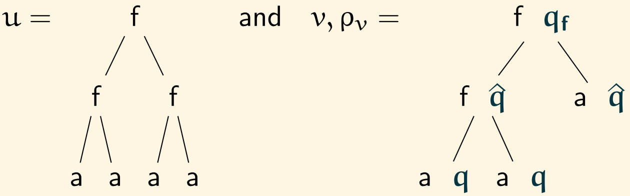


Tree Automata

With Global Equality Constraints [Filiot et al., 2008]

$\mathbb{A} = \{ \mathsf{a}/_0, \mathsf{f}/_2 \}$, $Q = \{ q, \hat{q}, q_f \}$, $F = \{ q_f \}$, $\hat{q} \approx \hat{q}$, and

$$\Delta = \{ \mathsf{f}(\hat{q}, \hat{q}) \to q_f,\ \mathsf{f}(q, q) \to q,\ \mathsf{f}(q, q) \to \hat{q},\ \mathsf{a} \to q,\ \mathsf{a} \to \hat{q} \}$$

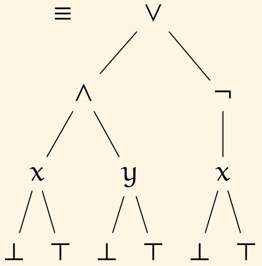


Tree Automata

With Global Equality Constraints [Filiot et al., 2008]

$\mathbb{A} = \{\wedge, \vee/_2, \neg/_1, \top, \bot/_0\} \uplus \mathbb{X}$, $Q = \{q_0, q_1\} \uplus \{v_x \mid x \in \mathbb{X}\}$ and $F = \{q_1\}$, new rules $\top \to v_x$, $\bot \to v_x$, $x(q_0, v_x) \to q_1$, $x(v_x, q_1) \to q_0$ for each $x \in \mathbb{X}$, $v_x \cong v_x$.

$(x \wedge y) \vee \neg x \quad \equiv$

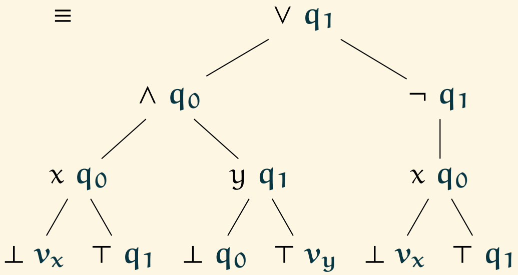


Tree Automata

With Global Equality Constraints [Filiot et al., 2008]

$\mathbb{A} = \{\wedge, \vee/2, \neg/1, \top, \bot/0\} \uplus \mathbb{X}$, $Q = \{q_0, q_1\} \uplus \{v_x \mid x \in \mathbb{X}\}$ and $F = \{q_1\}$, new rules $\top \to v_x$, $\bot \to v_x$, $x(q_0, v_x) \to q_1$, $x(v_x, q_1) \to q_0$ for each $x \in \mathbb{X}$, $v_x \approxeq v_x$.

$(x \wedge y) \vee \neg x \quad \equiv$



TA versus RTA versus $TA^=$

Closure, Complexity and Decidability

	TA	**RTA** $(p \approx p)$	$\mathbf{TA}^=$
$\cup$	PTIME	PTIME	PTIME
$\cap$	PTIME	EXPTIME	EXPTIME
$\neg$	EXPTIME	$\varnothing$	$\varnothing$
$t \in \mathcal{L}(\mathcal{A})$?	PTIME	NP-c	NP-c [(a)]
$\mathcal{L}(\mathcal{A}) = \varnothing$?	linear-time	linear-time	EXPTIME-c
$\lvert\mathcal{L}(\mathcal{A})\rvert \in \mathbb{N}$?	PTIME	PTIME	EXPTIME-c
$\mathcal{L}(\mathcal{A}) = \mathcal{T}(\Sigma)$?	EXPTIME-c	*undecidable*	*undecidable*
$\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{B})$?	EXPTIME-c	*undecidable*	*undecidable*
$\mathcal{L}(\bigcap_i \mathcal{A}_i) = \varnothing$?	EXPTIME-c	EXPTIME-c	EXPTIME-c

(a) SAT solver approach: [Héam et al., 2010].

TA versus RTA versus TA$^=$

Closure, Complexity and Decidability

	TA	**RTA** $(p \approx p)$	**TA$^=$**
$\cup$	PTime	PTime	PTime
$\cap$	PTime	ExpTime	ExpTime
$\neg$	ExpTime	$\varnothing$	$\varnothing$
$t \in \mathcal{L}(\mathcal{A})$?	PTime	NP-c	NP-c [a]
$\mathcal{L}(\mathcal{A}) = \varnothing$?	linear-time	**linear-time**	**ExpTime-c**
$\lvert\mathcal{L}(\mathcal{A})\rvert \in \mathbb{N}$?	PTime	**PTime**	**ExpTime-c**
$\mathcal{L}(\mathcal{A}) = \mathcal{T}(\Sigma)$?	ExpTime-c	*undecidable*	*undecidable*
$\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{B})$?	ExpTime-c	*undecidable*	*undecidable*
$\mathcal{L}(\bigcap_i \mathcal{A}_i) = \varnothing$?	ExpTime-c	ExpTime-c	ExpTime-c

[a] SAT solver approach: [Héam et al., 2010].

$\mathsf{TA}^=$ versus $\mathsf{TA}_k^=$

Restriction on the **kind** of constraints: Rigid Automata (RTA)

- Same expressive power as $\mathsf{TA}^=$
- Less compact representations
- Linear emptiness / finiteness tests, vs. ExpTime-complete
- Applications: [Jacquemard et al., 2009, Vacher, 2010]

What of the **number** of constraints? $\mathsf{TA}_k^=$ $\mathcal{A} = \langle \Sigma, Q, F, \Delta, \approxeq \rangle$:

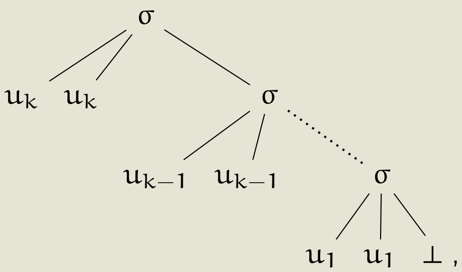
$\langle \Sigma, Q, F, \Delta, \approxeq \rangle$	$\mathsf{TA}^=$ $\mathcal{A}$
$\approxeq$	such that $\mathrm{Card}(\approxeq) \leqslant k$

Expressive Power

The Separation Languages $L = (\ell_k)_{k\in\mathbb{N}}$ [Hugot, 2013]

$$\biguplus_{i=1}^{k} \mathbb{A}_i \uplus \{\, \sigma/3, \bot/0 \,\} \qquad \mathbb{A}_i = \{\, a_i, b_i/0, f_i, g_i/2 \,\}$$

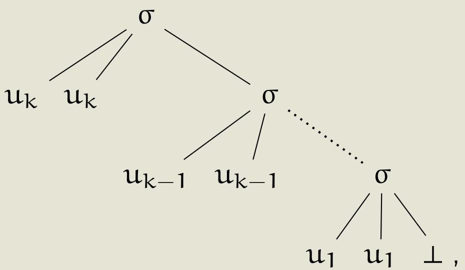
$$\ell_0 = \{\bot\} \quad \forall k \geqslant 1, \ell_k = \{\, \sigma(u, u, t_{k-1}) \mid u \in \mathcal{T}(\mathbb{A}_k), t_{k-1} \in \ell_{k-1} \,\}$$



$u_i \in \mathcal{T}(\mathbb{A}_i)\ \forall i$

Expressive Power

The Separation Languages $L = (\ell_k)_{k \in \mathbb{N}}$ [Hugot, 2013]



$u_i \in \mathcal{T}(\mathbb{A}_i)\ \forall i$

$\ell_1 \in \mathcal{L}(\mathsf{TA}_1^{\equiv}) \setminus \mathcal{L}(\mathsf{TA})$ $\approx$ ground instances of $f(x, x)$.

$\ell_k \in \mathcal{L}(\mathsf{TA}_k^{\equiv}) \setminus \mathcal{L}(\mathsf{TA}_{k-1}^{\equiv}), \quad \forall k \geqslant 1.$

Expressive Power

Show $\ell_k \in \mathcal{L}(\mathsf{TA}^{\equiv}_k) \setminus \mathcal{L}(\mathsf{TA}^{\equiv}_{k-1})$ [Hugot, 2013]

Show $\boldsymbol{\ell_k \in \mathcal{L}(\mathsf{TA}^{\equiv}_k)}$: $\mathcal{A}_k \in \mathsf{TA}^{\equiv}_k$ such that $\mathcal{L}(\mathcal{A}_k) = \ell_k$

$\mathcal{U}_i \in \mathsf{TA}$ universal, $\mathcal{U}_i{:}F = \{q^u_i\}$, for all i. $\mathcal{A}_k$ is

$$Q = \{q^v_0\} \uplus \biguplus_{i=1}^{k} \mathcal{U}_i{:}Q \uplus \{q^v_i\} \qquad F = \{q^v_1\} \qquad q^u_i \approxeq q^u_i,\ \forall i \in [\![1,k]\!]$$

$$\Delta = \left\{\, \sigma(q^u_i, q^u_i, q^v_{i-1}) \to q^v_i \;\middle|\; i \in [\![1,k]\!] \,\right\} \cup \{\, \bot \to q^v_0 \,\}\,.$$

Expressive Power

Show $\ell_k \in \mathcal{L}(\mathsf{TA}^{\equiv}_k) \setminus \mathcal{L}(\mathsf{TA}^{\equiv}_{k-1})$ [Hugot, 2013]

Show $\boldsymbol{\ell_k \notin \mathcal{L}(\mathsf{TA}^{\equiv}_{k-1})}$:

active constrained states:

$$\operatorname{acs}\rho = \{\, \rho(\alpha) \mid \alpha \in \mathcal{P}(\rho), \exists \beta \in \mathcal{P}(\rho) \setminus \{\alpha\} : \rho(\alpha) \approxeq \rho(\beta) \,\}$$

Expressive Power

Show $\ell_k \in \mathcal{L}(\mathsf{TA}^{\equiv}_{k}) \setminus \mathcal{L}(\mathsf{TA}^{\equiv}_{k-1})$ [Hugot, 2013]

Show $\ell_k \notin \mathcal{L}(\mathbf{TA}^{\equiv}_{k-1})$:

- Assume $\ell_k \in \mathcal{L}(\mathsf{TA}^{\equiv}_{k-1})$ i.e. $\exists \mathcal{A} \in \mathsf{TA}^{\equiv}_{k-1} : \mathcal{L}(\mathcal{A}) = \ell_k$

Expressive Power

Show $\ell_k \in \mathcal{L}(\mathsf{TA}_k^{\equiv}) \setminus \mathcal{L}(\mathsf{TA}_{k-1}^{\equiv})$ [Hugot, 2013]

Show $\ell_k \notin \mathcal{L}(\mathbf{TA}_{k-1}^{\equiv})$:

- Assume $\ell_k \in \mathcal{L}(\mathsf{TA}_{k-1}^{\equiv})$ i.e. $\exists \mathcal{A} \in \mathsf{TA}_{k-1}^{\equiv} : \mathcal{L}(\mathcal{A}) = \ell_k$
- $\forall \rho, \ \nexists \alpha, \beta : \alpha \neq \beta, \alpha \in 3^*, \rho(\alpha) \approxeq \rho(\beta)$

Expressive Power

Show $\ell_k \in \mathcal{L}(\mathrm{TA}_k^{\equiv}) \setminus \mathcal{L}(\mathrm{TA}_{k-1}^{\equiv})$ [Hugot, 2013]

Show $\ell_k \notin \mathcal{L}(\mathbf{TA}_{k-1}^{\equiv})$:

- Assume $\ell_k \in \mathcal{L}(\mathrm{TA}_{k-1}^{\equiv})$ i.e. $\exists \mathcal{A} \in \mathrm{TA}_{k-1}^{\equiv} : \mathcal{L}(\mathcal{A}) = \ell_k$
- $\forall \rho,\ \nexists \alpha, \beta : \alpha \neq \beta, \alpha \in 3^*, \rho(\alpha) \approxeq \rho(\beta)$
- Pick $t \in \ell_k$ such that $\left| |t|_\alpha \right| > |Q|$, for all $\alpha \in 3^*(1+2)$

Expressive Power

Show $\ell_k \in \mathcal{L}(\mathrm{TA}_k^{\equiv}) \setminus \mathcal{L}(\mathrm{TA}_{k-1}^{\equiv})$ [Hugot, 2013]

Show $\boldsymbol{\ell_k \notin \mathcal{L}(\mathrm{TA}_{k-1}^{\equiv})}$:

- Assume $\ell_k \in \mathcal{L}(\mathrm{TA}_{k-1}^{\equiv})$ i.e. $\exists \mathcal{A} \in \mathrm{TA}_{k-1}^{\equiv} : \mathcal{L}(\mathcal{A}) = \ell_k$
- $\forall \rho,\ \nexists \alpha, \beta : \alpha \neq \beta, \alpha \in 3^*, \rho(\alpha) \approxeq \rho(\beta)$
- Pick $t \in \ell_k$ such that $\left|t|_\alpha\right| > |Q|$, for all $\alpha \in 3^*(1+2)$
- Suppose $\exists \alpha \in 3^*(1+2)$ such that $\operatorname{ran} \rho|_\alpha \cap \operatorname{acs} \rho = \varnothing$. $\mathcal{A}$ acts as BUTA wrt. $t|_\alpha$; pump $\rho|_\alpha$, get $t' \notin \ell_k$, but $t' \in \mathcal{L}(\mathcal{A})$.

Expressive Power

Show $\ell_k \in \mathcal{L}(\mathrm{TA}_k^{\equiv}) \setminus \mathcal{L}(\mathrm{TA}_{k-1}^{\equiv})$ [Hugot, 2013]

Show $\ell_k \notin \mathcal{L}(\mathbf{TA}_{k-1}^{\equiv})$:

- Assume $\ell_k \in \mathcal{L}(\mathrm{TA}_{k-1}^{\equiv})$ i.e. $\exists \mathcal{A} \in \mathrm{TA}_{k-1}^{\equiv} : \mathcal{L}(\mathcal{A}) = \ell_k$
- $\forall \rho,\ \nexists \alpha, \beta : \alpha \neq \beta, \alpha \in 3^*, \rho(\alpha) \approxeq \rho(\beta)$
- Pick $t \in \ell_k$ such that $\left| t|_\alpha \right| > |Q|$, for all $\alpha \in 3^*(1+2)$
- $\forall \alpha \in 3^*(1+2),\ \operatorname{ran} \rho|_\alpha \cap \operatorname{acs} \rho \neq \varnothing$

Expressive Power

Show $\ell_k \in \mathcal{L}(\mathsf{TA}^{\equiv}_{k}) \setminus \mathcal{L}(\mathsf{TA}^{\equiv}_{k-1})$ [Hugot, 2013]

Show $\ell_k \notin \mathcal{L}(\mathbf{TA}^{\equiv}_{k-1})$:

- Assume $\ell_k \in \mathcal{L}(\mathsf{TA}^{\equiv}_{k-1})$ i.e. $\exists \mathcal{A} \in \mathsf{TA}^{\equiv}_{k-1} : \mathcal{L}(\mathcal{A}) = \ell_k$
- $\forall \rho,\ \nexists \alpha, \beta : \alpha \neq \beta, \alpha \in 3^*, \rho(\alpha) \approxeq \rho(\beta)$
- Pick $t \in \ell_k$ such that $\left| t|_\alpha \right| > |Q|$, for all $\alpha \in 3^*(1+2)$
- $\forall \alpha \in 3^*(1+2),\ \operatorname{ran} \rho|_\alpha \cap \operatorname{acs} \rho \neq \emptyset$
- $i \neq j$, p_i acs for u_i, p_j for u_j. $\exists$ acs $q_i, q_j : p_i \approxeq q_i, p_j \approxeq q_j$. Suppose q_i in subrun of u_j. Then $\exists s_i \trianglelefteq u_i, s_j \trianglelefteq u_j$, $s_i = s_j$. But $u_i \in \mathcal{T}(\mathbb{A}_i)$ and $u_j \in \mathcal{T}(\mathbb{A}_j)$, thus $s_i \in \mathcal{T}(\mathbb{A}_i)$ and $s_j \in \mathcal{T}(\mathbb{A}_j)$. $\mathcal{T}(\mathbb{A}_i) \cap \mathcal{T}(\mathbb{A}_j) = \emptyset$, thus $s_i = s_j \in \emptyset$.

Expressive Power

Show $\ell_k \in \mathcal{L}(\mathsf{TA}^{\equiv}_k) \setminus \mathcal{L}(\mathsf{TA}^{\equiv}_{k-1})$ [Hugot, 2013]

Show $\ell_k \notin \mathcal{L}(\mathbf{TA}^{\equiv}_{k-1})$:

- Assume $\ell_k \in \mathcal{L}(\mathsf{TA}^{\equiv}_{k-1})$ i.e. $\exists \mathcal{A} \in \mathsf{TA}^{\equiv}_{k-1} : \mathcal{L}(\mathcal{A}) = \ell_k$
- $\forall \rho,\ \nexists \alpha, \beta : \alpha \neq \beta, \alpha \in 3^*, \rho(\alpha) \approxeq \rho(\beta)$
- Pick $t \in \ell_k$ such that $\left| t|_\alpha \right| > |Q|$, for all $\alpha \in 3^*(1+2)$
- $\forall \alpha \in 3^*(1+2),\ \operatorname{ran} \rho|_\alpha \cap \operatorname{acs} \rho \neq \varnothing$
- Each pair of u_i needs its own fresh state(s) $p_i \approxeq q_i$

Expressive Power

Show $\ell_k \in \mathcal{L}(\mathrm{TA}^{\equiv}_{k}) \setminus \mathcal{L}(\mathrm{TA}^{\equiv}_{k-1})$ [Hugot, 2013]

Show $\ell_k \notin \mathcal{L}(\mathbf{TA}^{\equiv}_{k-1})$:

- Assume $\ell_k \in \mathcal{L}(\mathrm{TA}^{\equiv}_{k-1})$ i.e. $\exists \mathcal{A} \in \mathrm{TA}^{\equiv}_{k-1} : \mathcal{L}(\mathcal{A}) = \ell_k$
- $\forall \rho,\ \nexists \alpha, \beta : \alpha \neq \beta, \alpha \in 3^*, \rho(\alpha) \approxeq \rho(\beta)$
- Pick $t \in \ell_k$ such that $\left| t|_\alpha \right| > |Q|$, for all $\alpha \in 3^*(1+2)$
- $\forall \alpha \in 3^*(1+2),\ \operatorname{ran} \rho|_\alpha \cap \operatorname{acs} \rho \neq \varnothing$
- Each pair of u_i needs its own fresh state(s) $p_i \approxeq q_i$
- $\mathcal{A}$ does not exist, contradiction.

The Membership Problem

General Idea & Strategy

Membership complexity : $t \in \mathcal{L}(\mathcal{A})$?

NP-complete	for	$TA^{=}$
PTime	for	$TA^{=}_{k}, \forall k \in \mathbb{N}$

Proof **Strategy** :

- Choose each $P \subseteq \mathrm{dom} \approxeq = \{ p \mid \exists q : p \approxeq q \text{ or } q \approxeq p \}$
- Given P, turn $\approxeq$ into an equivalence relation $\asymp_P$
- Try all possible “housings” of the $\approxeq$-classes into t
- For each housing, try to build an accepting run

$\approxeq$ is Not an Equivalence

(but we can pretend it is)

Example: Given $p \approxeq r$ and $r \approxeq q$, what of $p \approxeq q$?

Does r actually appear in the run ?

yes : $p \approxeq q$ implied
no : $p \approxeq r$ and $r \approxeq q$ are moot.

Fix $P \subseteq \operatorname{dom} \approxeq$. Any run ρ such that $(\mathbf{ran}\,\rho) \cap (\mathbf{dom} \approxeq) = \mathbf{P}$ is accepting for $\mathcal{A}$ iff it is so for

$$\mathcal{A}_P = \wr \mathcal{A} \mid \approxeq := \left(\approxeq \cap P^2\right)^{\equiv} \int ,$$

symmetric, transitive, reflexive closure under $\operatorname{dom}(\approxeq \cap P^2)$.

Groups & Similarity Classes

Groups $\mathbb{G}_P$: set of $\approxeq$-equivalence classes (given P)

$$\mathbb{G}_P = \frac{\mathrm{dom}(\approxeq \cap P^2)}{(\approxeq \cap P^2)^{\equiv}} = \frac{\mathrm{dom}(\approxeq \cap P^2)}{\asymp_P}$$

Similarity **Classes** $\mathbb{S}_t$ of t :

$$\begin{aligned} \forall \alpha, \beta \in \mathcal{P}(t);\ \alpha \sim \beta \quad &\iff \quad t|_\alpha = t|_\beta \\ \textbf{classes } \mathbb{S}_t \quad &= \quad \mathcal{P}(t)/_\sim \end{aligned}$$

Housings

And Their Compatibility with the Constraints

Characterisation of **Satisfaction** of $\approxeq$:

$$\forall G \in \mathbb{G}_P;\ \exists C_G \in \mathbb{S}_t : \rho^{-1}(G) \subseteq C_G$$

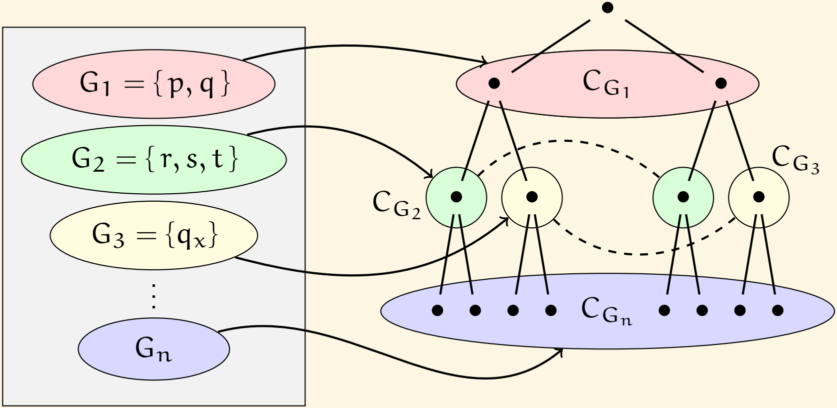
Housings $\mathbb{H}_P^t$ of P within t :

The map $G \mapsto C_G$ is a **P-housing of ρ in t**, **compatible** with ρ

$$\mathbb{H}_P^t = \mathbb{G}_P \to \mathbb{S}_t$$

is the set of all possible P-housings on t.

$\mathbb{G}_\mathbf{P}$ $\longleftarrow$ $\mathbf{h} \in \mathbb{H}_\mathbf{P}^\mathbf{t}$ $\longrightarrow$ $\mathbb{S}_\mathbf{t}$



Proof Outline

For $TA_k^=$

Operations Needed :

- **Choose P:** 2^{2k} possible $P \subseteq \mathrm{dom} \approxeq$
- **Choose housing:** $\left|\mathbb{S}_t{}^{\mathbb{G}_P}\right| = |\mathbb{S}_t|^{|\mathbb{G}_P|} \leqslant \|t\|^{2k}$ P-housings on t
- $\Rightarrow 4^k \cdot \|t\|^{2k}$ tests in total

↬ **polynomial** compatibility test = variant of **reachability**

Is a final state reachable if states $q \in P$ can only go in $h([q]_{\asymp_P})$?

Compatibility Test

In Polynomial Time

Simple variant of **reachability** algorithm:

Given P and $h \in \mathbb{H}_P^t$, there exists a compatible run iff

$$\Phi_t^{P,h}(\varepsilon) \cap F \neq \varnothing ,$$

where

$$\Phi_t^{P,h}(\alpha) = \left\{ q \in Q \;\middle|\; \begin{array}{l} t(\alpha)(p_1, \dots, p_n) \to q \in \Delta \\ \forall i \in [\![1, n]\!],\ p_i \in \Phi_t^{P,h}(\alpha.i) \\ q \in \bigcup \mathbb{G}_P \implies \alpha \in h([q]_{\asymp_P}) \\ q \notin \mathrm{dom}(\approxeq) \setminus P \end{array} \right\} .$$

Compatibility Test

In Polynomial Time

Simple variant of **reachability** algorithm:

Given P and $h \in \mathbb{H}_P^t$, there exists a compatible run iff

$$\Phi_t^{P,h}(\varepsilon) \cap F \neq \varnothing \ ,$$

where

$$\Phi_t^{P,h}(\alpha) = \left\{ q \in Q \;\middle|\; \begin{array}{l} t(\alpha)(p_1, \dots, p_n) \to q \in \Delta \\ \forall i \in [\![1, n]\!],\ p_i \in \Phi_t^{P,h}(\alpha.i) \\ q \in \bigcup \mathbb{G}_P \implies \alpha \in h([q]_{\asymp_P}) \\ q \notin \operatorname{dom}(\approxeq) \setminus P \end{array} \right\} .$$

Compatibility Test

In Polynomial Time

Simple variant of **reachability** algorithm:

Given P and $h \in \mathbb{H}_P^t$, there exists a compatible run iff

$$\Phi_t^{P,h}(\varepsilon) \cap F \neq \varnothing \ ,$$

where

$$\Phi_t^{P,h}(\alpha) = \left\{ q \in Q \;\middle|\; \begin{array}{l} t(\alpha)(p_1, \dots, p_n) \to q \in \Delta \\ \forall i \in [\![1, n]\!],\ p_i \in \Phi_t^{P,h}(\alpha.i) \\ q \in \bigcup \mathbb{G}_P \implies \alpha \in h([q]_{\asymp_P}) \\ q \notin \mathrm{dom}(\approxeq) \setminus P \end{array} \right\} .$$

Compatibility Test

In Polynomial Time

Simple variant of **reachability** algorithm:

Given P and $h \in \mathbb{H}_P^t$, there exists a compatible run iff

$$\Phi_t^{P,h}(\varepsilon) \cap F \neq \varnothing \ ,$$

where

$$\Phi_t^{P,h}(\alpha) = \left\{ q \in Q \ \middle| \begin{array}{l} t(\alpha)(p_1, \dots, p_n) \to q \in \Delta \\ \forall i \in [\![1, n]\!],\ p_i \in \Phi_t^{P,h}(\alpha.i) \\ q \in \bigcup \mathbb{G}_P \implies \alpha \in h([q]_{\asymp_P}) \\ q \notin \mathrm{dom}(\approxeq) \setminus P \end{array} \right\} .$$

Compatibility Test

In Polynomial Time

Simple variant of **reachability** algorithm:

Given P and $h \in \mathbb{H}_P^t$, there exists a compatible run iff

$$\Phi_t^{P,h}(\varepsilon) \cap F \neq \varnothing \ ,$$

where

$$\Phi_t^{P,h}(\alpha) = \left\{ q \in Q \;\middle|\; \begin{array}{l} t(\alpha)(p_1, \dots, p_n) \to q \in \Delta \\ \forall i \in [\![1, n]\!],\ p_i \in \Phi_t^{P,h}(\alpha.i) \\ q \in \bigcup \mathbb{G}_P \implies \alpha \in h([q]_{\asymp_P}) \\ q \notin \mathrm{dom}(\approxeq) \setminus P \end{array} \right\} .$$

Rigidification

Problem : Given $TA^=$ $\mathcal{A}$, build equivalent RTA $\mathcal{B}$.

General Result [Filiot, 2008, Lem. 5.3.5]

Exponential construction: $\|\mathcal{B}\| \leqslant O(2^{\|\mathcal{A}\|^2})$

In the case of $\mathbf{TA}_1^=$:

Polynomial construction: $\|\mathcal{B}\| \leqslant O(\|\mathcal{A}\|^2)$

Idea : Simulate a constraint $p \approxeq q$, $p \neq q$ by a TA intersection

Rigidification: Construction

$$\mathcal{B} = \mathcal{B}_p^\neg \uplus \mathcal{B}_q^\neg \uplus \wr \mathcal{A} \mid Q', \Delta', q_f \approxeq q_f \int$$

$$\begin{aligned}
\mathcal{B}_p^\neg &= \wr \mathcal{A} \mid Q \setminus \{p\} \int & \mathcal{B}_q^\neg &= \wr \mathcal{A} \mid Q \setminus \{q\} \int \\
Q' &= (Q \setminus \{p, q\}) \uplus (\mathcal{B}_{pq} : Q) & \Delta' &= \Delta_{pq}^{q_f} \uplus (\mathcal{B}_{pq} : \Delta) \\
\mathcal{B}_{pq} &= \mathcal{B}_p \otimes \mathcal{B}_q & q_f &= (p, q) \\
\mathcal{B}_p &= \wr \mathcal{B}_q^\neg \mid F := \{p\}, \Delta := \Delta_p \int & \mathcal{B}_q &= \wr \mathcal{B}_p^\neg \mid F := \{q\}, \Delta := \Delta_q \int \\
\Delta_p &= \mathcal{B}_q^\neg : \Delta \setminus \{ \dots p \dots \to \dots \} & \Delta_q &= \mathcal{B}_p^\neg : \Delta \setminus \{ \dots q \dots \to \dots \}
\end{aligned}$$

$\Delta_{pq}^{q_f}$ is $\mathcal{A} : \Delta$ from which all left-hand side occurrences of p or q have been replaced by q_f.

Emptiness

Outline of the Result and Proof

Complexity of **Emptiness** : $\mathcal{L}(\mathcal{A}) = \varnothing$?

PTime (quadratic)	for	$\mathrm{TA}_1^{\equiv}$
ExpTime-complete	for	$\mathrm{TA}_k^{\equiv}$, $k \geqslant 2$

$\mathrm{TA}_1^{\equiv}$: immediate by **rigidification**. Emptiness for RTA: linear time

$\mathrm{TA}_2^{\equiv}$: Reduction of **intersection-emptiness** of n TA $\mathcal{A}_1, \dots, \mathcal{A}_n$.
Generalisation of the usual argument [Filiot et al., 2008, Thm. 1] from “unlimited constraints” to **“two constraints”**

$$L = \varnothing \iff \bigcap_{i=1}^{n} \mathcal{L}(\mathcal{A}_i) = \varnothing$$

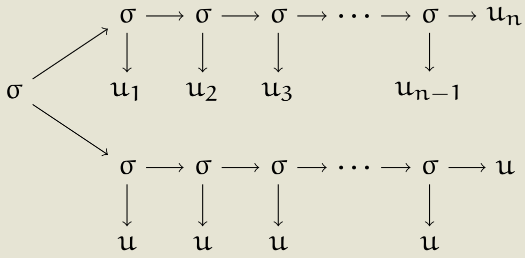


Figure : Reduction of intersection-emptiness: the language.

where $\forall i,\ x_i \in \mathcal{L}(\mathcal{A}_i)$ and $x = x_i$

Finiteness

Outline of the Result and Proof

Complexity of **Finiteness** : $|\mathcal{L}(\mathcal{A})| \in \mathbb{N}$?

PTime	for	$\mathsf{TA}_1^{\equiv}$
ExpTime-complete	for	$\mathsf{TA}_k^{\equiv}$, $k \geqslant 2$

$\mathsf{TA}_1^{\equiv}$: immediate by **rigidification**. Finiteness for RTA is PTime

$\mathsf{TA}_2^{\equiv}$: Reduction of **Emptiness for $\mathsf{TA}_2^{\equiv}$**.

Finiteness

Outline of the Result and Proof

$$\mathcal{A}' = \wr\, \mathcal{A} \mid Q \uplus \{p\}, F := \{p\}, \Sigma \uplus \{\sigma/1\}, \Delta' \,\wr$$
$$\text{where } \Delta' = \Delta \cup \{\, \sigma(q_f) \to p \mid q_f \in F \,\} \cup \{\, \sigma(p) \to p \,\}$$

$$\begin{array}{rcl} \text{if } \mathcal{L}(\mathcal{A}) = \varnothing & \textbf{then} & \mathcal{L}(\mathcal{A}') = \varnothing \\ \text{if } t \in \mathcal{L}(\mathcal{A}) & \textbf{then} & \sigma^+(t) \subseteq \mathcal{L}(\mathcal{A}') \end{array}$$

$\mathcal{L}(\mathcal{A}')$ is **finite** $\iff$ $\mathcal{L}(\mathcal{A})$ is **empty**

Summary

and Perspectives

Refined **complexity** and **expressiveness** results:

- **Expressiveness:** $\mathsf{TA}_k^=$ form a strict hierarchy
- **Membership:** NP-c for $\mathsf{TA}^=$, but PTime for $\mathsf{TA}_k^=$, $\forall k$
- **Emptiness:** quadratic for $\mathsf{TA}_1^=$, ExpTime-complete for $\mathsf{TA}_2^=$
- **Finiteness:** PTime for $\mathsf{TA}_1^=$, ExpTime-complete for $\mathsf{TA}_2^=$

Left **to do**:

Effects of $\not\cong$, flat constraints, efficient heuristics, etcetera.

Tree Walking Automata

in a Few Words

- **Not** a **new** formalism [Aho and Ullman, 1969]
- **Sequential** model, as opposed to branching tree automata
- **Less** extensively **studied** model until ≈ 2000
- [Bojańczyk and Colcombet, 2005, Bojańczyk and Colcombet, 2006]
- Recent **surge in interest**, due mostly to connection to **XML**:
  - Caterpillar expressions [Brüggemann-Klein and Wood, 2000]
  - Streaming XML documents [Segoufin and Vianu, 2002]
  - type-checking XML-QL, XSLT,. . . [Milo et al., 2003]
- Rich **variants**: pebbles, marbles,. . .

Tree Walking Automata

in a Few Words

Existing research focused on **fundamental** problems: expressive power, determinisability,. . .

We study practical, efficient **algorithms**

In particular: the transformation from **TWA to BUTA**

Preliminaries

Definition of Tree Walking Automata

A **Tree-Walking Automaton** is a tuple $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$

$$\Delta \subseteq \Sigma \times Q \times \underbrace{\{\star, \mathbf{0}, \mathbf{1}\}}_{\mathbb{T}\ :\ \text{types}} \times \underbrace{\{\uparrow, \circlearrowleft, \swarrow, \searrow\}}_{\mathbb{M}\ :\ \text{moves}} \times Q$$

- "$\langle f, p, \tau \to \mu, q \rangle$" written for the tuple $(f, p, \tau, \mu, q) \in \Delta$.
- $\langle \Sigma_2, p, \mathbb{T} \to \circlearrowleft, q \rangle = \{ (\sigma, p, \tau, \circlearrowleft, q) \mid \sigma \in \Sigma_2, \tau \in \mathbb{T} \}$

Remarks

- Ranked (binary) vs. unranked alphabet
- $\langle \Sigma_0, Q, \mathbb{T} \to \{\swarrow, \searrow\}, Q \rangle \cup \langle \Sigma, Q, \star \to \uparrow, Q \rangle$ invalid

Preliminaries

Example Tree Walking Automaton

A very simple **example** TWA: $\mathcal{X} = \langle \Sigma, Q, I, F, \Delta \rangle$

- $\Sigma_0 = \{ a, b, c \}$ and $\Sigma_2 = \{ f, g, h \}$
- $Q = \{ q_\ell, q_u \}$, $I = \{q_\ell\}$, $F = \{q_u\}$

$$\Delta = \langle a, q_\ell, \{\star, \mathbf{0}\} \to \circlearrowleft, q_u \rangle \cup \langle \Sigma, q_u, \mathbf{0} \to \uparrow, q_u \rangle \cup \langle \Sigma_2, q_\ell, \{\star, \mathbf{0}\} \to \swarrow, q_\ell \rangle$$

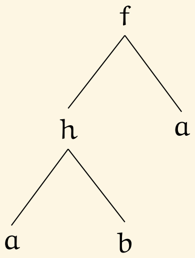
$\mathcal{X}$ accepts exactly all trees whose left-most leaf is labelled by a — and the tree a itself.

Preliminaries

Example Tree Walking Automaton

$Q = \{ q_\ell, q_u \}$, $\mathbf{I = \{q_\ell\}}$, $F = \{q_u\}$

$$\Delta = \langle a, q_\ell, \{\star, \mathbf{0}\} \to \circlearrowleft, q_u \rangle \cup \langle \Sigma, q_u, \mathbf{0} \to \uparrow, q_u \rangle \cup \langle \Sigma_2, q_\ell, \{\star, \mathbf{0}\} \to \swarrow, q_\ell \rangle$$

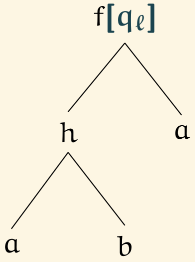


Preliminaries

Example Tree Walking Automaton

$Q = \{ q_\ell, q_u \}$, $I = \{q_\ell\}$, $F = \{q_u\}$

$$\Delta = \langle a, q_\ell, \{\star, \mathbf{0}\} \to \circlearrowleft, q_u \rangle \cup \langle \Sigma, q_u, \mathbf{0} \to \uparrow, q_u \rangle \cup \langle \boldsymbol{\Sigma_2}, \boldsymbol{q_\ell}, \{\star, \mathbf{0}\} \to \swarrow, \boldsymbol{q_\ell} \rangle$$

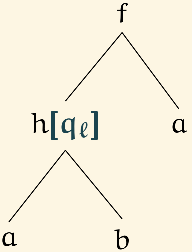


Preliminaries

Example Tree Walking Automaton

$Q = \{ q_\ell, q_u \}$, $I = \{q_\ell\}$, $F = \{q_u\}$

$$\Delta = \langle a, q_\ell, \{\star, \mathbf{0}\} \to \circlearrowleft, q_u \rangle \cup \langle \Sigma, q_u, \mathbf{0} \to \uparrow, q_u \rangle \cup \langle \Sigma_2, q_\ell, \{\star, \mathbf{0}\} \to \swarrow, q_\ell \rangle$$

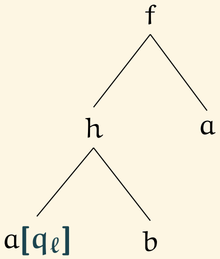


Preliminaries

Example Tree Walking Automaton

$Q = \{ q_\ell, q_u \}$, $I = \{q_\ell\}$, $F = \{q_u\}$

$$\Delta = \langle a, q_\ell, \{ \star, \mathbf{0} \} \to \circlearrowleft, q_u \rangle \cup \langle \Sigma, q_u, \mathbf{0} \to \uparrow, q_u \rangle \cup \langle \Sigma_2, q_\ell, \{ \star, \mathbf{0} \} \to \swarrow, q_\ell \rangle$$

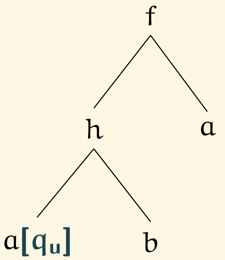


Preliminaries

Example Tree Walking Automaton

$Q = \{ q_\ell, q_u \}$, $I = \{q_\ell\}$, $F = \{q_u\}$

$$\Delta = \langle a, q_\ell, \{\star, \mathbf{0}\} \to \circlearrowleft, q_u \rangle \cup \langle \Sigma, q_u, \mathbf{0} \to \uparrow, q_u \rangle \cup \langle \Sigma_2, q_\ell, \{\star, \mathbf{0}\} \to \swarrow, q_\ell \rangle$$

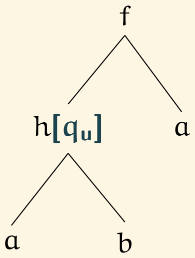


Preliminaries

Example Tree Walking Automaton

$Q = \{\, q_\ell, q_u \,\},\ I = \{q_\ell\},\ F = \{q_u\}$

$$\Delta = \langle a, q_\ell, \{\star, \mathbf{0}\} \to \circlearrowleft, q_u \rangle \cup \langle \boldsymbol{\Sigma}, \mathbf{q_u}, \mathbf{0} \to \uparrow, \mathbf{q_u} \rangle \cup \langle \Sigma_2, q_\ell, \{\star, \mathbf{0}\} \to \swarrow, q_\ell \rangle$$

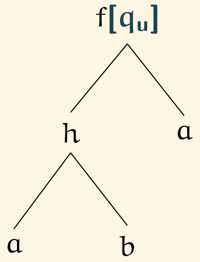


Preliminaries

Example Tree Walking Automaton

$Q = \{ q_\ell, q_u \}$, $I = \{q_\ell\}$, $F = \{q_u\}$

$$\Delta = \langle a, q_\ell, \{\star, \mathbf{0}\} \to \circlearrowleft, q_u \rangle \cup \langle \Sigma, q_u, \mathbf{0} \to \uparrow, q_u \rangle \cup \langle \Sigma_2, q_\ell, \{\star, \mathbf{0}\} \to \swarrow, q_\ell \rangle$$



TWA to BUTA Transformation

Given a TWA $\mathcal{A}$, build an equivalent BUTA $\mathcal{B}$

- **Solution** outlined in [Bojańczyk, 2008] and [Samuelides, 2007]
- Based on the idea of **tree loops**
- Claims resulting states for $\mathcal{B}$: $\mathbb{T} \times 2^{Q^2}$ — or det. $(2^{Q^2})^{\mathbb{T}}$

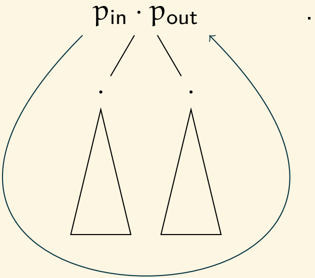
- Only **proof sketches**. No explicit algorithm is given.
- We argue that things are slightly less straightforward:
  - Needed **states space**: $\Sigma \times \mathbb{T} \times 2^{Q^2}$ — or det. $\Sigma \times (2^{Q^2})^{\mathbb{T}}$
  - Existing implementations: *almost* correct [dtwa-tools]
- We introduce **tree overloops**
  - This time we **really** have $\mathbb{T} \times 2^{Q^2}$ — or det. $(2^{Q^2})^{\mathbb{T}}$
  - **Nicer upper bound** if $\mathcal{A}$ is deterministic: $|\mathbb{T}| \cdot 2^{|Q| \log_2(|Q|+1)}$

The Idea of Tree Loops

With Pretty Pictures

$(p_{in}, p_{out}) \in Q^2$ is a **loop of $\mathcal{A}$ on $t|_\alpha$** if there exists a run which

- starts in p_{in},
- ends in p_{out} — at the local root α,
- and always stays in the subtree

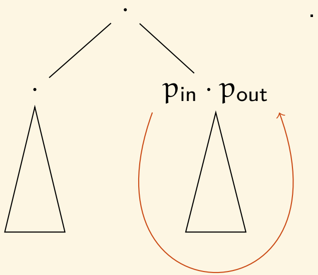


The Idea of Tree Loops

With Pretty Pictures

$(p_{in}, p_{out}) \in Q^2$ is a **loop of $\mathcal{A}$ on $t|_\alpha$** if there exists a run which

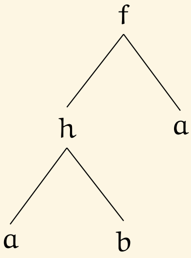
- starts in p_{in},
- ends in p_{out} — at the local root α,
- and **always stays in the subtree**



The Idea of Tree Loops

By Example

Recall that $\mathcal{X}$ visits the **left-most leaf** and goes back up if it is a.



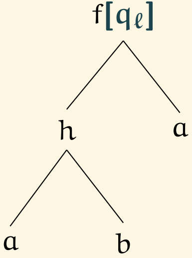
Loops of $\mathcal{X}$ on...

- t: $\{\}$
- $t|_0$: $\{\}$
- $t|_{0.0}$: $\{\}$
- $t|_{0.1}$: $\{\}$
- $t|_1$: $\{\}$

The Idea of Tree Loops

By Example

Recall that $\mathcal{X}$ visits the **left-most leaf** and goes back up if it is a.



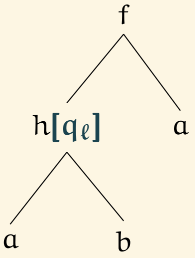
Loops of $\mathcal{X}$ on...

- t: $\{(q_\ell, ?), (q_\ell, q_\ell)\}$
- $t|_0$: $\{\}$
- $t|_{0.0}$: $\{\}$
- $t|_{0.1}$: $\{\}$
- $t|_1$: $\{\}$

The Idea of Tree Loops

By Example

Recall that $\mathcal{X}$ visits the **left-most leaf** and goes back up if it is a.



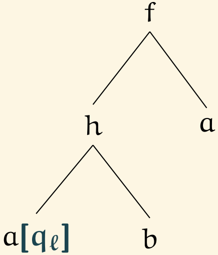
Loops of $\mathcal{X}$ on...

- t: $\{(q_\ell, ?), (q_\ell, q_\ell)\}$
- $t|_0$: $\{(q_\ell, ?), (q_\ell, q_\ell)\}$
- $t|_{0.0}$: $\{\}$
- $t|_{0.1}$: $\{\}$
- $t|_1$: $\{\}$

The Idea of Tree Loops

By Example

Recall that $\mathcal{X}$ visits the **left-most leaf** and goes back up if it is a.



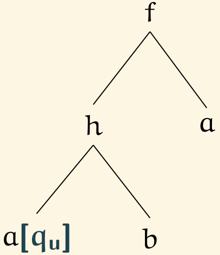
Loops of $\mathcal{X}$ on...

- t: $\{(q_\ell, \boldsymbol{?}), (q_\ell, q_\ell)\}$
- $t|_0$: $\{(q_\ell, \boldsymbol{?}), (q_\ell, q_\ell)\}$
- $t|_{0.0}$: $\{(q_\ell, \boldsymbol{?}), \mathbf{(q_\ell, q_\ell)}\}$
- $t|_{0.1}$: $\{\}$
- $t|_1$: $\{\}$

The Idea of Tree Loops

By Example

Recall that $\mathcal{X}$ visits the **left-most leaf** and goes back up if it is a.



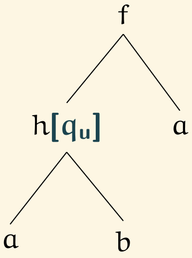
Loops of $\mathcal{X}$ on...

- t: $\{(q_\ell, \boldsymbol{?}), (q_\ell, q_\ell)\}$
- $t|_0$: $\{(q_\ell, \boldsymbol{?}), (q_\ell, q_\ell)\}$
- $t|_{0.0}$: $\{(q_\ell, q_u), (q_\ell, q_\ell), \mathbf{(q_u, q_u)}\}$
- $t|_{0.1}$: $\{\}$
- $t|_1$: $\{\}$

The Idea of Tree Loops

By Example

Recall that $\mathcal{X}$ visits the **left-most leaf** and goes back up if it is a.



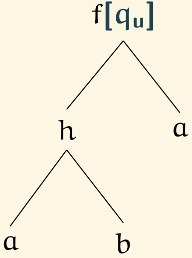
Loops of $\mathcal{X}$ on...

- t: $\{(q_\ell, \boldsymbol{?}), (q_\ell, q_\ell)\}$
- $t|_0$: $\{(q_\ell, \mathbf{q_u}), (q_\ell, q_\ell), \mathbf{(q_u, q_u)}\}$
- $t|_{0.0}$: $\{(q_\ell, q_u), (q_\ell, q_\ell), (q_u, q_u)\}$
- $t|_{0.1}$: $\{\}$
- $t|_1$: $\{\}$

The Idea of Tree Loops

By Example

Recall that $\mathcal{X}$ visits the **left-most leaf** and goes back up if it is a.



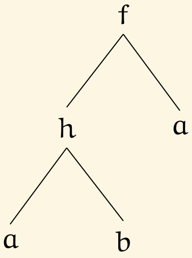
Loops of $\mathcal{X}$ on...

- t: $\{(q_\ell, q_u), (q_\ell, q_\ell), (q_u, q_u)\}$
- $t|_0$: $\{(q_\ell, q_u), (q_\ell, q_\ell), (q_u, q_u)\}$
- $t|_{0.0}$: $\{(q_\ell, q_u), (q_\ell, q_\ell), (q_u, q_u)\}$
- $t|_{0.1}$: $\{\}$
- $t|_1$: $\{\}$

The Idea of Tree Loops

By Example

Recall that $\mathcal{X}$ visits the **left-most leaf** and goes back up if it is a.



Loops of $\mathcal{X}$ on...

- t: $\{(q_\ell, q_u), (q_\ell, q_\ell), (q_u, q_u)\}$
- $t|_0$: $\{(q_\ell, q_u), (q_\ell, q_\ell), (q_u, q_u)\}$
- $t|_{0.0}$: $\{(q_\ell, q_u), (q_\ell, q_\ell), (q_u, q_u)\}$
- $t|_{0.1}$: $\{(q_\ell, q_\ell), (q_u, q_u)\}$
- $t|_1$: $\{(q_\ell, q_\ell), (q_u, q_u)\}$

Computing Tree Loops

Loops Decomposition

A loop is a **simple loop** on $t|_\alpha$ if there is a run which forms it and reaches α exactly twice — i.e. *simple looping run*

Proposition: **loops decomposition**

If $S \subseteq Q^2$ is the set of all simple loops of $\mathcal{A}$ on a given subtree $u = t|_\alpha$, then S^* is the set of all loops of $\mathcal{A}$ on u.

So to compute all loops, it **suffices** to compute **simple loops**.

Computing Tree Loops

$\mho^\tau(u)$ = set of loops of $\mathcal{A}$ on a subtree u of type τ

On **leaves** $u = a \in \Sigma_0$

Simple looping run $= (\alpha, p) \twoheadrightarrow (\alpha, q)$ only.

$$\mathcal{H}^\tau_\sigma = \{\, (p, q) \mid \langle \sigma, p, \tau \to \circlearrowleft, q \rangle \in \Delta \,\} \qquad \mho^\tau(a) = (\mathcal{H}^\tau_a)^*$$

On **inner nodes** $u = f(u_{\mathbf{0}}, u_{\mathbf{1}})$: by **first move**

- $\uparrow$ — impossible: leaves the subtree u
- $\circlearrowleft$ — all computed in $\mathcal{H}^\tau_f$
- $\swarrow$ — $(\varepsilon, p), (0, p_{\mathbf{0}}), (\beta_1, s_1), \dots, (\beta_n, s_n), (0, q_{\mathbf{0}}), (\varepsilon, q)$, with all $\beta_k \trianglelefteq 0$. So $(p_{\mathbf{0}}, q_{\mathbf{0}}) \in \mho^{\mathbf{0}}(u_{\mathbf{0}})$
- $\searrow$ —$(\varepsilon, p), (1, p_{\mathbf{1}}), (\beta_1, s_1), \dots, (\beta_n, s_n), (1, q_{\mathbf{1}}), (\varepsilon, q)$, with all $\beta_k \trianglelefteq 1$. So $(p_{\mathbf{1}}, q_{\mathbf{1}}) \in \mho^{\mathbf{1}}(u_{\mathbf{1}})$

Computing Tree Loops

$\mho^\tau(u)$ = set of loops of $\mathcal{A}$ on a subtree u of type τ

On **leaves** $u = a \in \Sigma_0$

Simple looping run $= (\alpha, p) \twoheadrightarrow (\alpha, q)$ only.

$$\mathcal{H}_\sigma^\tau = \{\,(p,q) \mid \langle \sigma, p, \tau \to \circlearrowleft, q\rangle \in \Delta\,\} \qquad \mho^\tau(a) = (\mathcal{H}_a^\tau)^*$$

On **inner nodes** $u = f(u_\mathbf{0}, u_\mathbf{1})$

1. choose a side: $\theta \in \mathbb{S} = \{\,\mathbf{0}, \mathbf{1}\,\}$
2. find an existing loop on that side: $(p_\theta, q_\theta) \in \mho^\theta(u_\theta)$
3. such that one can connect beginning and end
   1. $\langle f, p, \tau \to \chi(\theta), p_\theta\rangle \in \Delta^a$ and
   2. $\langle u_\theta(\varepsilon), q_\theta, \theta \to \uparrow, q\rangle \in \Delta$

---

[a] $\chi(\cdot) : \mathbb{S} \to \{\swarrow, \searrow\}$ such that $\chi(\mathbf{0}) = \swarrow$ and $\chi(\mathbf{1}) = \searrow$

Computing Tree Loops

$\mho^\tau(u)$ = set of loops of $\mathcal{A}$ on a subtree u of type τ

On **leaves** $u = a \in \Sigma_0$

Simple looping run $= (\alpha, p) \twoheadrightarrow (\alpha, q)$ only.

$$\mathcal{H}^\tau_\sigma = \{ (p, q) \mid \langle \sigma, p, \tau \to ⟲, q \rangle \in \Delta \} \qquad \mho^\tau(a) = (\mathcal{H}^\tau_a)^*$$

On **inner nodes** $u = f(u_0, u_1)$

$$\left(\mathcal{H}^\tau_f \cup \left\{ (p, q) \;\middle|\; \begin{array}{c} \exists \theta \in \mathbb{S} : \\ \exists (p_\theta, q_\theta) \in \mho^\theta(u_\theta) \end{array} : \begin{array}{c} \langle f, p, \tau \to \chi(\theta), p_\theta \rangle \in \Delta \\ \langle u_\theta(\varepsilon), q_\theta, \theta \to \uparrow, q \rangle \in \Delta \end{array} \right\} \right)^*$$

Loops-Based Transformation Into BUTA

0. **Input:** A TWA $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$
1. **Initialise** *States* and *Rules* to $\varnothing$
2. **for each** $a \in \Sigma_0, \tau \in \mathbb{T}$ **do**
   - let $P = (a, \tau, {\mathcal{H}_a^\tau}^*)$
     add $a \to P$ to *Rules* and P to *States*
3. **repeat until** *Rules* remain unchanged
   - **for each** $f \in \Sigma_2, \tau \in \mathbb{T}$ **do**
     - add every $f(P_\mathbf{0}, P_\mathbf{1}) \to P$ to *Rules* and P to *States* where $P_\mathbf{0}, P_\mathbf{1} \in$ *States* such that $P_\mathbf{0} = (\sigma_\mathbf{0}, \mathbf{0}, S_\mathbf{0})$ and $P_\mathbf{1} = (\sigma_\mathbf{1}, \mathbf{1}, S_\mathbf{1})$ and $P = (f, \tau, (\mathcal{H}_f^\tau \cup S)^*)$,
       with S the set of simple loops built on the sons.
4. **Output:** A BUTA $\mathcal{B}$ equivalent to $\mathcal{A}$:
   $\mathcal{B} = \langle \Sigma, \mathit{States}, \{\, (\sigma, \star, L) \in \mathit{States} \mid L \cap (I \times F) \neq \varnothing \,\}, \mathit{Rules} \rangle$

Loops-Based Transformation Into BUTA

0. **Input:** A TWA $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$
1. **Initialise** *States* and *Rules* to $\varnothing$
2. **for each** $a \in \Sigma_0, \tau \in \mathbb{T}$ **do**
   - let $P = (a, \tau, \mathcal{H}_a^{\tau *})$
     add $a \to P$ to *Rules* and P to *States*
3. **repeat until** *Rules* remain unchanged
   - **for each** $f \in \Sigma_2, \tau \in \mathbb{T}$ **do**
     - add every $f(P_\mathbf{0}, P_\mathbf{1}) \to P$ to *Rules* and P to *States* where $P_\mathbf{0}, P_\mathbf{1} \in$ *States* such that $P_\mathbf{0} = (\sigma_\mathbf{0}, \mathbf{0}, S_\mathbf{0})$ and $P_\mathbf{1} = (\sigma_\mathbf{1}, \mathbf{1}, S_\mathbf{1})$ and $P = (f, \tau, (\mathcal{H}_f^\tau \cup S)^*)$,
       with S the set of simple loops built on the sons.
4. **Output:** A BUTA $\mathcal{B}$ equivalent to $\mathcal{A}$:
   $\mathcal{B} = \langle \Sigma, \mathit{States}, \{ (\sigma, \star, L) \in \mathit{States} \mid L \cap (I \times F) \neq \varnothing \}, \mathit{Rules} \rangle$

Loops-Based Transformation Into BUTA

add every $f(P_\mathbf{0}, P_\mathbf{1}) \to P$ to *Rules* and P to *States*
where $P_\mathbf{0}, P_\mathbf{1} \in$ *States* such that

- $P_\mathbf{0} = (\sigma_\mathbf{0}, \mathbf{0}, S_\mathbf{0})$ and $P_\mathbf{1} = (\sigma_\mathbf{1}, \mathbf{1}, S_\mathbf{1})$
- and $P = (f, \tau, (\mathcal{H}_f^\tau \cup S)^*)$,
- with S the set of simple loops built on the sons.

$$S = \left\{ (p, q) \;\middle|\; \begin{array}{c} \exists\theta \in \mathbb{S} : \\ \exists(p_\theta, q_\theta) \in S_\theta \end{array} : \begin{array}{l} \langle f, p, \tau \to \chi(\theta), p_\theta \rangle \in \Delta \\ \langle \sigma_\theta, q_\theta, \theta \to \uparrow, q \rangle \in \Delta \end{array} \right\}$$

Loops-Based Transformation Into BUTA

add every $f(P_0, P_1) \to P$ to *Rules* and P to *States*
where $P_0, P_1 \in$ *States* such that

- $P_0 = (\sigma_0, \mathbf{0}, S_0)$ and $P_1 = (\sigma_1, \mathbf{1}, S_1)$
- and $P = (f, \tau, (\mathcal{H}_f^\tau \cup S)^*)$,
- with S the set of simple loops built on the sons.

$$S = \left\{ (p, q) \;\middle|\; \begin{array}{c} \exists\theta \in \mathbb{S} : \\ \exists(p_\theta, q_\theta) \in S_\theta \end{array} : \begin{array}{l} \langle f, p, \tau \to \chi(\theta), p_\theta \rangle \in \Delta \\ \langle \sigma_\theta, q_\theta, \theta \to \uparrow, q \rangle \in \Delta \end{array} \right\}$$

The **son's symbol** is needed to close the end of the loop!

Loops-Based Transformation Into BUTA

The Real States Space

Sets of loops **cannot** be considered independently from the **symbol** in which they are rooted.

Consider $\langle\{a, b\}, p, \tau \to \circlearrowleft, q\rangle$ and $\langle b, q, \tau \to \uparrow, s'\rangle \in \Delta$. Then $\mho^{\theta}(a) = \mho^{\theta}(b) = \{(p, q)\}^*$, but $\mho^{\tau}(f(a, a)) \neq \mho^{\tau}(f(b, b))$.

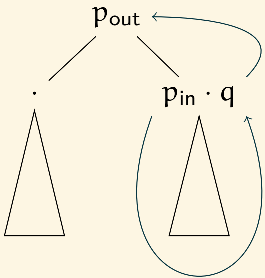
Needs states in $\Sigma \times \mathbb{T} \times 2^{Q^2}$ instead of just $\mathbb{T} \times 2^{Q^2}$.

Alphabet potentially large. **How to get rid of it ?**

From Tree Loops to Tree Overloops

Tree **overloops**: slight alteration of loops, with advantages.

- Fixes **states space**: $\mathbb{T} \times 2^{Q^2}$ instead of $\Sigma \times \mathbb{T} \times 2^{Q^2}$.
- Deterministic case: $|\mathbb{T}| \cdot 2^{|Q| \log_2(|Q|+1)}$ **better upper bound**
- 2 to 100 times **smaller BUTA** in average in random tests.



From Tree Loops to Tree Overloops

Tree **overloops**: slight alteration of loops, with advantages.

- Fixes **states space**: $\mathbb{T} \times 2^{Q^2}$ instead of $\Sigma \times \mathbb{T} \times 2^{Q^2}$.
- Deterministic case: $|\mathbb{T}| \cdot 2^{|Q|\log_2(|Q|+1)}$ **better upper bound**
- 2 to 100 times **smaller BUTA** in average in random tests.

$(p, q) \in Q^2$ is an **overloop** of $\mathcal{A}$ on $t|_\alpha$ if there exists a run which starts in p, ends in q at the *parent* of the root α, and always stays in the subtree, except for the last configuration.

Parent of ε is $\overline{\varepsilon}$. A TWA $\mathcal{A}$ must be **escaped** into
$\mathcal{A}' = \left\langle\ \Sigma,\ Q \uplus \{\checkmark\},\ I, F,\ \Delta \uplus \langle \Sigma, F, \star \to \uparrow, \checkmark \rangle\ \right\rangle$.

Overloops and Determinism

A TWA $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$ is **deterministic** if for all $\sigma \in \Sigma, p \in Q, \tau \in \mathbb{T}, \ |\langle \sigma, p, \tau \to \mathbb{M}, Q \rangle \cap \Delta| \leqslant 1$.

In general, the overloops-based BUTA has up to $|\mathbb{T}| \times 2^{|Q|^2}$ states. However, it has at most $|\mathbb{T}| \cdot 2^{|Q| \log_2(|Q|+1)}$ states if $\mathcal{A}$ is a DTWA.

If $\mathcal{A}$ is deterministic, **overloop sets are functional**. Not like loops.

Partial functions versus relations.

At most $|Q + 1|^{|Q|}$ overloop sets, versus $2^{|Q|^2}$.

Polynomial Approximation for Emptiness

Emptiness is ExpTime-complete

- XML Queries / Caterpillar accessibility
- Satisfiability of some XPath fragments
- But also TWA model-checking...

Standard: TWA $\to$ BUTA (explosion) $\to$ linear test. Alternative:

- An **over-approximation**; *may* detect emptiness
- Polynomial time and space
- Very – surprisingly – accurate in our random tests

Polynomial Approximation for Emptiness

0. **Input:** An *escaped* TWA $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$
1. **Initialise** $\mathcal{L}_{\mathbf{0}}$, $\mathcal{L}_{\mathbf{1}}$, $\mathcal{L}_{\star}$ to $\varnothing$
2. **for each** $a \in \Sigma_0, \tau \in \mathbb{T}$ **do**
   - $\mathcal{L}_\tau \leftarrow \mathcal{L}_\tau \cup \mathcal{U}_a^\tau[{\mathcal{H}_a^\tau}^*]$
3. **repeat until** $\mathcal{L}_{\mathbf{0}}$, $\mathcal{L}_{\mathbf{1}}$, $\mathcal{L}_{\star}$ remain unchanged
   - **for each** $f \in \Sigma_2, \tau \in \mathbb{T}$ **do**
     - $\mathcal{L}_\tau \leftarrow \mathcal{L}_\tau \cup \mathcal{U}_f^\tau[(\mathcal{H}_f^\tau \cup S)^*]$
       with S the set of simple loops built on $\mathcal{L}_{\mathbf{0}}$ and $\mathcal{L}_{\mathbf{1}}$.
4. **Output:** *Empty* if $\mathcal{L}_\star \cap (I \times \{\checkmark\}) = \varnothing$, else *Unknown*

Polynomial Approximation for Emptiness

0. **Input:** An *escaped* TWA $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$
1. **Initialise** $\mathcal{L}_\mathbf{0}$, $\mathcal{L}_\mathbf{1}$, $\mathcal{L}_\star$ to $\varnothing$
2. **for each** $a \in \Sigma_0, \tau \in \mathbb{T}$ **do**
   - $\mathcal{L}_\tau \leftarrow \mathcal{L}_\tau \cup \mathcal{U}_a^\tau[\mathcal{H}_a^{\tau *}]$
3. **repeat until** $\mathcal{L}_\mathbf{0}$, $\mathcal{L}_\mathbf{1}$, $\mathcal{L}_\star$ remain unchanged
   - **for each** $f \in \Sigma_2, \tau \in \mathbb{T}$ **do**
     - $\mathcal{L}_\tau \leftarrow \mathcal{L}_\tau \cup \mathcal{U}_f^\tau[(\mathcal{H}_f^\tau \cup S)^*]$
       with S the set of simple loops built on $\mathcal{L}_\mathbf{0}$ and $\mathcal{L}_\mathbf{1}$.
4. **Output:** *Empty* if $\mathcal{L}_\star \cap (I \times \{\checkmark\}) = \varnothing$, else *Unknown*

coarsest with one bucket $\mathcal{L}$; **finest** as full transformation (exp)

Polynomial Approximation for Emptiness

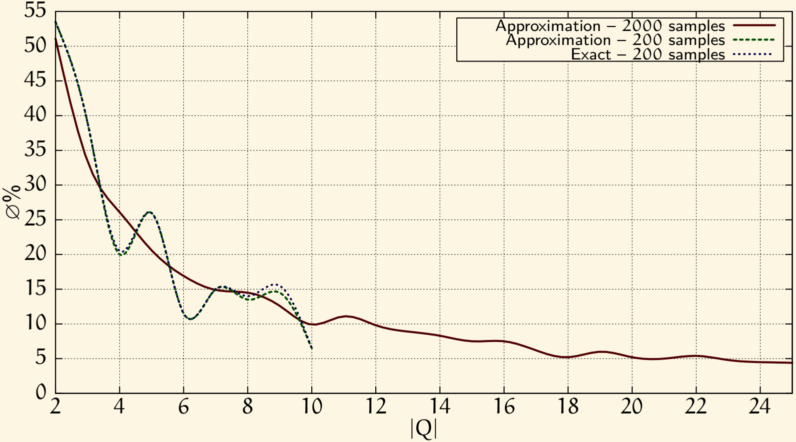
Random tests

1. **Ad-hoc** scheme: $\approx 20\,000$ TWA, $2 \leqslant |Q| \leqslant 20$, $|\Delta| \approx 3 \times |Q|$, 75% of empty languages, only two *Unknown* instead of *Empty*.

2. **Uniform** scheme [Héam et al., 2009], REGAL back-end for FSA generation [Bassino et al., 2007]. $2\,000$ deterministic and complete TWA uniformly generated for each $2 \leqslant |Q| \leqslant 25$.

Polynomial Approximation for Emptiness

Random tests



Size Comparison: Loops vs. Overloops

One Example & Uniform Generation Scheme

For $\mathcal{X}$: loops $\|\mathcal{B}_\mathsf{l}\| = 1986$; overloops $\|\mathcal{B}_\mathsf{o}\| = 95$; deterministic minimal $\|\mathcal{B}_\mathsf{m}\| = 56$; smallest known non-deterministic $\|\mathcal{B}_\mathsf{s}\| = 34$.

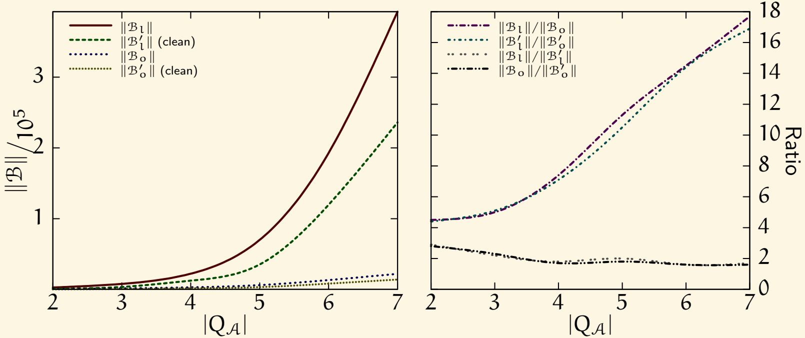
Loops **60 times** worse than manual optimal; overloops **3 times**.

Orthogonal to **post-processing** cleanup: $\|\mathcal{B}_\mathsf{l}'\| = 1617$, $\|\mathcal{B}_\mathsf{o}'\| = 78$.

$$\frac{\|\mathcal{B}_\mathsf{l}\|}{\|\mathcal{B}_\mathsf{o}\|} \approx 20.9 \quad \text{and} \quad \frac{\|\mathcal{B}_\mathsf{l}'\|}{\|\mathcal{B}_\mathsf{o}'\|} \approx 20.7 \quad \text{and} \quad \frac{\|\mathcal{B}_\mathsf{l}\|}{\|\mathcal{B}_\mathsf{l}'\|} \approx \frac{\|\mathcal{B}_\mathsf{o}\|}{\|\mathcal{B}_\mathsf{o}'\|} \approx 1.2\ .$$

Size Comparison: Loops vs. Overloops

One Example & Uniform Generation Scheme



References I

Aho, A. and Ullman, J. (1969).
Translations on a context free grammar.
Information and Control, 19(5):439–475.

Bassino, F., David, J., and Nicaud, C. (2007).
REGAL : A library to randomly and exhaustively generate automata.
In *CIAA, LNCS 4783*, pages 303–305.

Bojańczyk, M. (2008).
Tree-Walking Automata.
LATA'08 (tutorial), LNCS, 5196.

Bojańczyk, M. and Colcombet, T. (2005).
Tree-walking automata do not recognize all regular languages.
STOC '05, pages 234–243. ACM.

References II

Bojańczyk, M. and Colcombet, T. (2006).
Tree-walking automata cannot be determinized.
Theoretical Computer Science, 350(2-3):164–173.

Bouajjani, A. and Touili, T. (2002).
Extrapolating tree transformations.
In Brinksma, E. and Larsen, K. G., editors, *Computer Aided Verification, CAV'02*, volume 2404 of *Lecture Notes in Computer Science*, pages 539–554. Springer-Verlag.

Boyer, B. and Genet, T. (2009).
Verifying Temporal Regular Properties of Abstractions of Term Rewriting Systems.
In *RULE*, volume 21 of *EPTCS*, pages 99–108.

References III

- Brüggemann-Klein, A. and Wood, D. (2000).
  Caterpillars: A context specification technique.
  Markup Languages, 2(1):81–106.

- Clarke, E. M. and Emerson, E. A. (1981).
  Design and synthesis of synchronization skeletons using branching-time temporal logic.
  In Kozen, D., editor, *Logic of Programs*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71. Springer.

- Comon, H., Dauchet, M., Gilleron, R., Löding, C., Jacquemard, F., Lugiez, D., Tison, S., and Tommasi, M. (2008).
  Tree Automata Techniques and Applications.
  release November 18, 2008.

References IV

Courbis, R., Héam, P.-C., and Kouchnarenko, O. (2009).
TAGED Approximations for Temporal Properties Model-Checking.
In [Maneth, 2009], pages 135–144.

Dwyer, M., Avrunin, G., and Corbett, J. (1999).
Patterns in property specifications for finite-state verification.
In *ICSE'99*, pages 411–420. IEEE.

Feuillade, G., Genet, T., and Tong, V. V. T. (2004).
Reachability analysis over term rewriting systems.
J. Autom. Reasoning, 33(3-4):341–383.

References V

Filiot, E. (2008).
Logics for n-ary queries in trees.
PhD thesis, Université des Sciences et Technologie de Lille - Lille I.

Filiot, E., Talbot, J.-M., and Tison, S. (2008).
Tree automata with global constraints.
In *Developments in Language Theory*, volume 5257 of *Lecture Notes in Computer Science*, pages 314–326. Springer.

Héam, P., Hugot, V., and Kouchnarenko, O. (2010).
SAT Solvers for Queries over Tree Automata with Constraints.
In *Third International Conference on Software Testing, Verification, and Validation Workshops*, pages 343–348. IEEE.

References VI

Héam, P.-C., Hugot, V., and Kouchnarenko, O. (2011).
Loops and overloops for tree walking automata.
In *CIAA'11, LNCS 6807*, pages 166–177.

Héam, P.-C., Hugot, V., and Kouchnarenko, O. (2012a).
From linear temporal logic properties to rewrite propositions.
In Gramlich, B., Miller, D., and Sattler, U., editors, *IJCAR'12*, volume 7364 of *Lecture Notes in Computer Science*, pages 316–331. Springer.

Héam, P.-C., Hugot, V., and Kouchnarenko, O. (2012b).
Loops and overloops for tree-walking automata.
Theoretical Computer Science, 450:43–53.

References VII

Héam, P.-C., Hugot, V., and Kouchnarenko, O. (2012c).
On positive TAGED with a bounded number of constraints.
In Moreira, N. and Reis, R., editors, *CIAA*, volume 7381 of *Lecture Notes in Computer Science*, pages 329–336. Springer.

Héam, P.-C., Nicaud, C., and Schmitz, S. (2009).
Random generation of deterministic tree (walking) automata.
In [Maneth, 2009], pages 115–124.

Hugot, V. (2013).
Tree Automata, Approximations, and Constraints for Verification – Tree (Not Quite) Regular Model-Checking.
Ph.D. thesis (to be defended), Université de Franche-Comté.

References VIII

Jacquemard, F., Klay, F., and Vacher, C. (2009).
Rigid tree automata.
In Horia Dediu, A., Mihai Ionescu, A., and Martín-Vide, C., editors, *Proceedings of the 3rd International Conference on Language and Automata Theory and Applications (LATA'09)*, volume 5457 of *Lecture Notes in Computer Science*, pages 446–457, Tarragona, Spain. Springer.

Kesten, Y., Maler, O., Marcus, M., Pnueli, A., and Shahar, E. (1997).
Symbolic model checking with rich assertional languages.
In Grumberg, O., editor, *CAV*, volume 1254 of *Lecture Notes in Computer Science*, pages 424–435. Springer.

References IX

Maneth, S., editor (2009).
Implementation and Application of Automata, 14th International Conference, CIAA 2009, Sydney, Australia, July 14-17, 2009. Proceedings, volume 5642 of *Lecture Notes in Computer Science*. Springer.

Manna, Z. and Pnueli, A. (1995).
Temporal Verification of Reactive Systems - Safety.
Springer.

Meseguer, J. (1992).
Conditioned rewriting logic as a unified model of concurrency.
Theoretical Computer Science, 96(1):73–155.

References X

Milo, T., Suciu, D., and Vianu, V. (2003).
Typechecking for XML transformers.
J. Comput. Syst. Sci., 66(1):66–97.

Queille, J.-P. and Sifakis, J. (1982).
Specification and verification of concurrent systems in CESAR.
In Dezani-Ciancaglini, M. and Montanari, U., editors, *Symposium on Programming*, volume 137 of *Lecture Notes in Computer Science*, pages 337–351. Springer.

Samuelides, M. (2007).
Automates d'arbres à jetons.
PhD thesis, Université Paris-Diderot - Paris VII.

References XI

Segoufin, L. and Vianu, V. (2002).
Validating Streaming XML Documents.
In *PODS*, pages 53–64. ACM.

Vacher, C. (2010).
Tree automata with global constraints for the verification of security properties.
Ph.D. thesis, ENS Cachan.