Loops and Overloops for Tree Walking Automata

Pierre-Cyrille Héam, Vincent Hugot, Olga Kouchnarenko

{pcheam,vhugot,okouchnarenko}@lifc.univ-fcomte.fr

University of Franche-Comté
DGA & LIFC-INRIA/CASSIS, project ACCESS

July 13, 2011

Tree Walking Automata

- Old formalism (≈1970, Aho & Ullman)
- Sequential model, as opposed to *branching* tree automata
- Less extensively studied model for a long while...
- ...but long standing questions solved in recent years
- Recent surge in interest, due mostly to connection to XML (Fragments of Core XPath, streaming etc)

- Research focused on fundamental problems (expressiveness...)
- *Our focus: practical, efficient algorithms*
- *Finality: efficient XML queries; compact Model-Checking*

Starting Point:

Transformation from TWA to BUTA

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

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

Notations

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

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

A Very Simple TWA: $\mathcal{X} = \langle \Sigma, Q, I, F, \Delta \rangle$

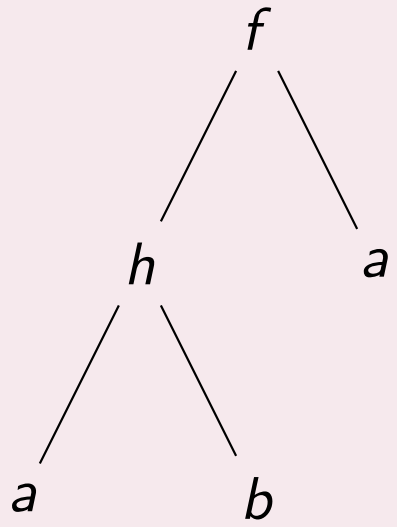
- $\Sigma_0 = \{ a, b, c \}$ and $\Sigma_2 = \{ f, g, h \}$
- $Q = \{ q_\ell, q_u \}$, $I = \{q_\ell\}$, $F = \{q_u\}$

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

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

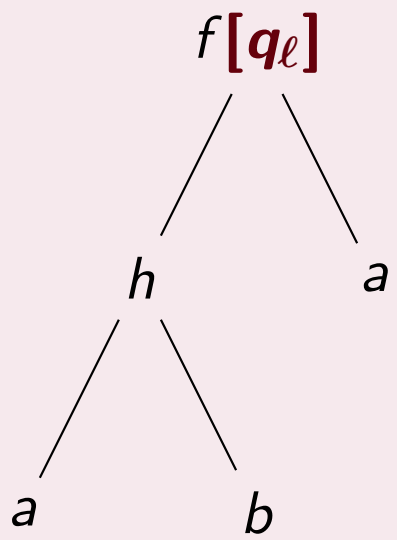
$Q = \{ q_\ell, q_u \}$, $I = \{q_\ell\}$, $F = \{q_u\}$

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



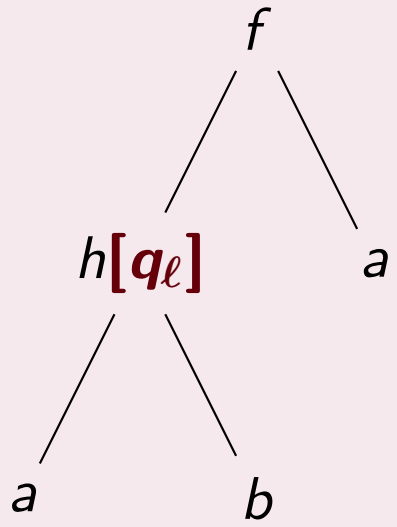
$Q = \{ q_\ell, q_u \}$, $I = \{q_\ell\}$, $F = \{q_u\}$

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



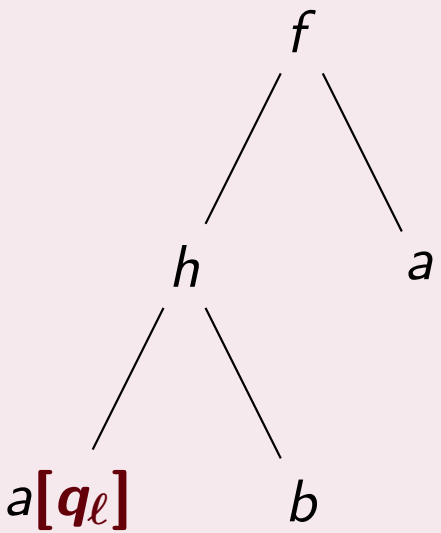
$Q = \{ q_\ell, q_u \}$, $I = \{q_\ell\}$, $F = \{q_u\}$

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



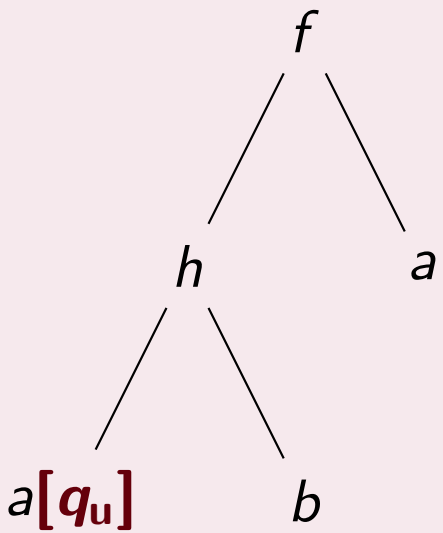
$Q = \{ q_\ell, q_u \}$, $I = \{q_\ell\}$, $F = \{q_u\}$

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



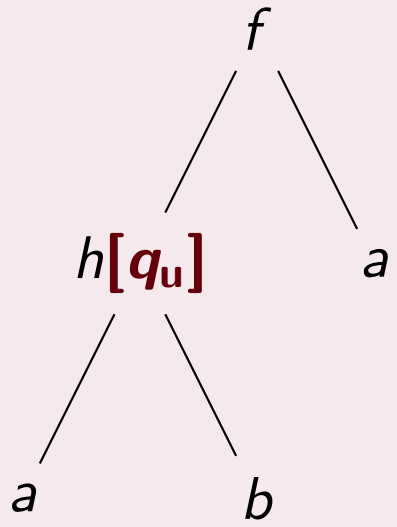
$Q = \{ q_\ell, q_u \}$, $I = \{q_\ell\}$, $F = \{q_u\}$

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



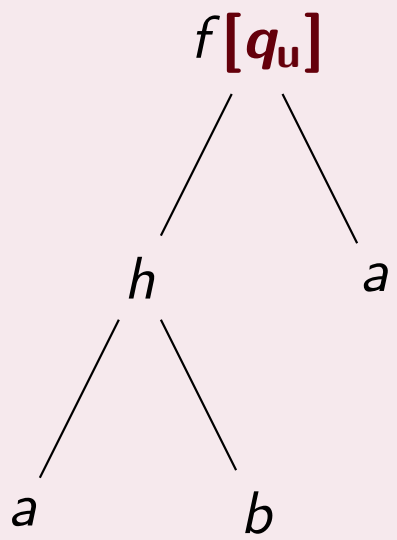
$Q = \{ q_\ell, q_u \}$, $I = \{q_\ell\}$, $F = \{q_u\}$

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



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

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



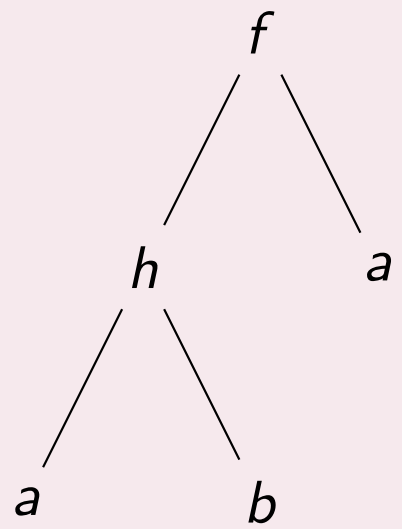
Given a TWA $\mathcal{A}$, build an equivalent BUTA $\mathcal{B}$.

- Known solution outlined in the literature [Bojańczyk, Samuelides]
- Based on the idea of **tree loops**
- 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. $\Sigma \times (2^{Q^2})^{\mathbb{T}}$)
  - Because of this, some existing implementations are only *almost* correct [dtwa-tools]
- We introduce **tree overloops**
  - This time we really have $\mathbb{T} \times 2^{Q^2}$ (or det. $(2^{Q^2})^{\mathbb{T}}$)
  - Lower upper bound if $\mathcal{A}$ is deterministic: $|\mathbb{T}| \cdot 2^{|Q| \log_2(|Q|+1)}$

The Idea of Tree Loops

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

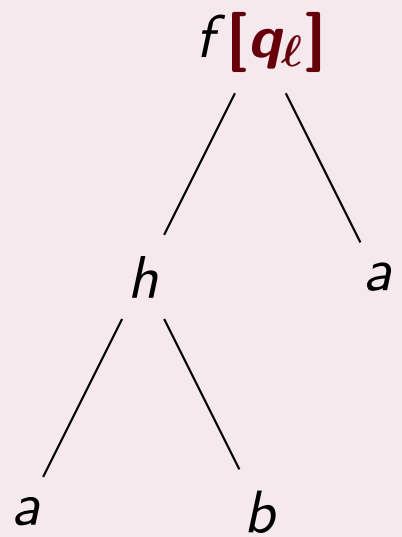


Loops of $\mathcal{X}$ on...

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

The Idea of Tree Loops

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

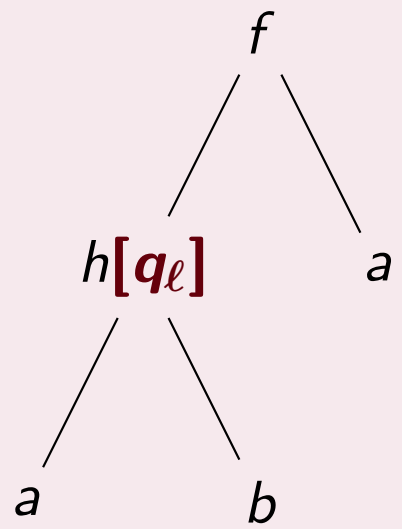


Loops of $\mathcal{X}$ on...

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

The Idea of Tree Loops

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

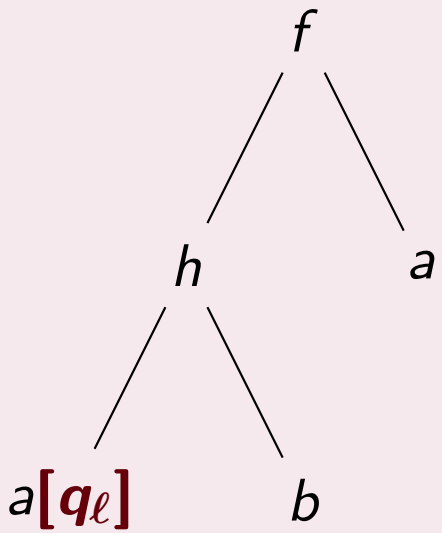


Loops of $\mathcal{X}$ on...

- 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

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

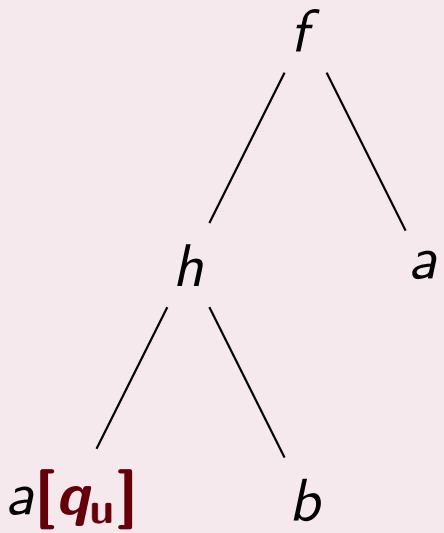


Loops of $\mathcal{X}$ on...

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

The Idea of Tree Loops

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

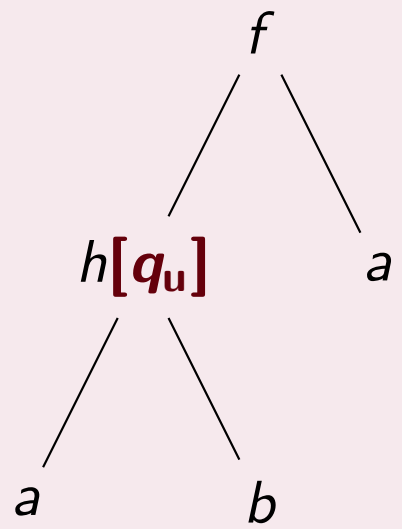


Loops of $\mathcal{X}$ on...

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

The Idea of Tree Loops

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

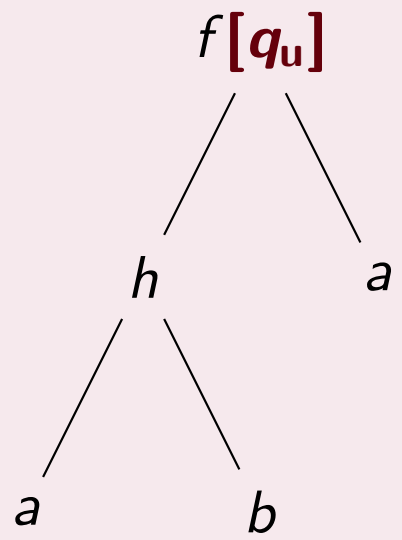


Loops of $\mathcal{X}$ on...

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

The Idea of Tree Loops

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

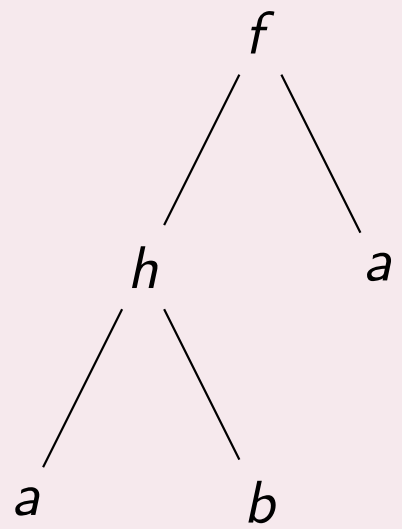


Loops of $\mathcal{X}$ on...

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

The Idea of Tree Loops

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



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

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

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

We denote $\mathcal{U}^\tau(u)$ the set of all loops of $\mathcal{A}$ on a subtree u, where τ is the type of the root of u. Compute loops on $u = a \in \Sigma_0$.

Simple looping run: run of the form $(\varepsilon, p) \twoheadrightarrow (\varepsilon, q)$ only.

$$\mathcal{H}^\tau_\sigma \stackrel{\text{def}}{=} \{\, (p,q) \mid \langle \sigma, p, \tau \to \circlearrowleft, q \rangle \in \Delta \,\}.$$

So we have $\mathcal{U}^\tau(a) = (\mathcal{H}^\tau_a)^*$.

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

We denote $\mathcal{U}^\tau(u)$ the set of all loops of $\mathcal{A}$ on a subtree u, where τ is the type of the root of u. Compute loops on $u = a \in \Sigma_0$.

Simple looping run: run of the form $(\varepsilon, p) \twoheadrightarrow (\varepsilon, q)$ only.

$$\mathcal{H}^\tau_\sigma \stackrel{\text{def}}{=} \{\, (p,q) \mid \langle \sigma, p, \tau \to \circlearrowleft, q \rangle \in \Delta \,\}.$$

So we have $\mathcal{U}^\tau(a) = (\mathcal{H}^\tau_a)^*$.

Let $f \in \Sigma_2$, and $u = f(u_0, u_1)$; root of type τ. Compute $\mathcal{U}^\tau(u)$.

First Move of a Simple Looping Run

- $\uparrow$ — impossible: leaves the subtree u
- $\circlearrowleft$ — all computed in $\mathcal{H}_f^\tau$
- $\swarrow$ — $(\varepsilon, p), (0, p_0), (\beta_1, s_1), \ldots, (\beta_n, s_n), (0, q_0), (\varepsilon, q)$, with all $\beta_k \trianglelefteq 0$. So $(p_0, q_0) \in \mathcal{U}^0(u_0)$.
- $\searrow$ —$(\varepsilon, p), (1, p_1), (\beta_1, s_1), \ldots, (\beta_n, s_n), (1, q_1), (\varepsilon, q)$, with all $\beta_k \trianglelefteq 1$. So $(p_1, q_1) \in \mathcal{U}^1(u_1)$.

Computing Tree Loops
Computation for Inner Nodes

Let $f \in \Sigma_2$, and $u = f(u_{\boldsymbol{0}}, u_{\boldsymbol{1}})$; root of type τ. Compute $\mho^\tau(u)$.

First Move of a Simple Looping Run

- $\swarrow$ — $(\varepsilon, p), (0, p_{\boldsymbol{0}}), (\beta_1, s_1), \ldots, (\beta_n, s_n), (0, q_{\boldsymbol{0}}), (\varepsilon, q)$, with all $\beta_k \trianglelefteq 0$. So $(p_{\boldsymbol{0}}, q_{\boldsymbol{0}}) \in \mho^{\boldsymbol{0}}(u_{\boldsymbol{0}})$.

To build a simple loop (p, q) on the subtree u, we need to...

1. choose a side: $\theta \in \mathbb{S} \stackrel{\text{def}}{=} \{\boldsymbol{0}, \boldsymbol{1}\}$
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 $\chi(\boldsymbol{0}) = \swarrow$ and $\chi(\boldsymbol{1}) = \searrow$.

Loops on Leaves

Let $a \in \Sigma_0$, we have $\mho^\tau(a) = (\mathcal{H}_a^\tau)^*$.

Loops on Inner Nodes

Let $f \in \Sigma_2$, and $u = f(u_0, u_1)$; root of type τ. We have $\mho^\tau(u) =$

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

Transformation Into BUTA Using Loops

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 $f(P_0, P_1) \to P$ 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.
4. **Output:** A BUTA $\mathcal{B}$ equivalent to $\mathcal{A}$:
   $\mathcal{B} = \langle \Sigma, States, \{ (\sigma, \star, L) \in States \mid L \cap (I \times F) \neq \varnothing \}, Rules \rangle$

Transformation Into BUTA Using Loops

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 $f(P_0, P_1) \to P$ 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.
4. **Output:** A BUTA $\mathcal{B}$ equivalent to $\mathcal{A}$:
   $\mathcal{B} = \langle \Sigma, States, \{ (\sigma, \star, L) \in States \mid L \cap (I \times F) \neq \varnothing \}, Rules \rangle$

Transformation Into BUTA

Using Loops

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.

Set of Simple Loops Built on the Sons

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

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

Transformation Into BUTA

Using Loops

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.

Set of Simple Loops Built on the Sons

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

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

Transformation Into BUTA

Using Loops

Important Remark

In the construction, sets of loops **cannot** be considered independently from the symbol they are rooted in.

Counter-Example

Consider $a, b \in \Sigma_0, f \in \Sigma_2$, with only the transitions

$$\langle \{ a, b \}, p, \theta \to \circlearrowleft, q \rangle \cup \{ \langle b, q, \theta \to \uparrow, s' \rangle, \langle f, s, \tau \to \chi(\theta), p \rangle \} \subseteq \Delta$$

Then $\mho^{\theta}(a) = \mho^{\theta}(b) = \{(p, q)\}^*$, but $\mho^{\tau}(f(a, a)) \neq \mho^{\tau}(f(b, b))$.

Thus the loops-based construction has $\Sigma \times \mathbb{T} \times 2^{Q^2}$ instead of only $\mathbb{T} \times 2^{Q^2}$ states (storing the symbol).

Tree Overloops: slight alteration of the notion of *tree loop*.

Advantages wrt. Transformation Into Buta

- Straightforward $\mathbb{T} \times 2^{Q^2}$ instead of $\Sigma \times \mathbb{T} \times 2^{Q^2}$ states
- DTWA $\mathcal{A}$: smaller $|\mathbb{T}| \cdot 2^{|Q| \log_2(|Q|+1)}$ upper bound on states
- 2 to 100 times smaller BUTA in average (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 $\mathcal{A}' = \langle\, \Sigma,\ Q \uplus \{\checkmark\},\ I, F,\ \Delta \uplus \langle \Sigma, F, \star \to \uparrow, \checkmark \rangle \,\rangle$.

Tree Overloops: slight alteration of the notion of *tree loop*.

Advantages wrt. Transformation Into Buta

- Straightforward $\mathbb{T} \times 2^{Q^2}$ instead of $\Sigma \times \mathbb{T} \times 2^{Q^2}$ states
- DTWA $\mathcal{A}$: smaller $|\mathbb{T}| \cdot 2^{|Q| \log_2(|Q|+1)}$ upper bound on states
- 2 to 100 times smaller BUTA in average (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 $\mathcal{A}' = \langle\ \Sigma,\ Q \uplus \{\checkmark\},\ I, F,\ \Delta \uplus \langle \Sigma, F, \star \to \uparrow, \checkmark \rangle\ \rangle$.

Idea: Compute loops, then check for $\uparrow$-transitions

Definition: Up-Closure

Let $L \subseteq Q^2, \tau \in \mathbb{T}$ and $\sigma \in \Sigma$:

$$\mathcal{U}_\sigma^\tau[L] \stackrel{\text{def}}{=} \{ (p,q) \mid \exists p' : (p,p') \in L \text{ and } \langle \sigma, p', \tau \to \uparrow, q \rangle \in \Delta \} \quad .$$

Theorem: Up-Closure

Let $\mathcal{A}$ be a TWA. If L is the set of all loops of $\mathcal{A}$ on a subtree $u = t|_\alpha$, then $\mathcal{U}_{t(\alpha)}^{\natural\alpha}[L]$ is the set of all overloops of $\mathcal{A}$ on u.

Overloops on Leaves

Let $a \in \Sigma_0$, we have $\mathfrak{O}^\tau(a) = \mathcal{U}_a^\tau[\mathcal{H}_a^{\tau *}]$.

Overloops on Inner Nodes

Let $f \in \Sigma_2$, and $u = f(u_{\boldsymbol{0}}, u_{\boldsymbol{1}})$; root of type τ. To build a *loop* (p, q_θ) on a subtree, we need to...

1. choose a side: $\theta \in \mathbb{S} \stackrel{\text{def}}{=} \{\, \boldsymbol{0}, \boldsymbol{1} \,\}$
2. find an existing overloop on that side: $(p_\theta, q_\theta) \in \mathfrak{O}^\theta(u_\theta)$
3. such that one can connect the beginning:
   1. $\langle f, p, \tau \to \chi(\theta), p_\theta \rangle \in \Delta^a$
   2. *unlike loops, the end is already conected!*

---

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

Overloops on Leaves

Let $a \in \Sigma_0$, we have $\overleftarrow{\mathfrak{O}}^{\tau}(a) = \mathcal{U}_a^{\tau}[\mathcal{H}_a^{\tau *}]$.

Overloops on Inner Nodes

Let $f \in \Sigma_2$, and $u = f(u_0, u_1)$; root of type τ. We have $\overleftarrow{\mathfrak{O}}^{\tau}(u) =$

$$\mathcal{U}_f^{\tau}\left[\left(\mathcal{H}_f^{\tau} \cup \left\{ (p, q_\theta) \;\middle|\; \begin{array}{l} \exists \theta \in \mathbb{S} : \\ \exists p_\theta \in Q \end{array} : \begin{array}{l} \langle f, p, \tau \to \chi(\theta), p_\theta \rangle \in \Delta \\ \text{and } (p_\theta, q_\theta) \in \overleftarrow{\mathfrak{O}}^{\theta}(u_\theta) \end{array} \right\}\right)^*\right]$$

Transformation Into BUTA Using Overloops

Almost the same the the loops-based version, but:

1. We compute sets of *overloops* instead of *loops*
2. Symbols are not stored in the states (not needed): thus $\Sigma \times \mathbb{T} \times 2^{Q^2}$ becomes $\mathbb{T} \times 2^{Q^2}$ again.
3. Final states are

$$\{ (\star, O) \in States \mid O \cap (I \times \{\checkmark\}) \neq \varnothing \}$$

Acceptance criterion: Final Loops & Overloops

A term t is accepted iff

- $\mho^\tau(t) \cap I \times F \neq \varnothing$
- $\overline{\mho}^\tau(t) \cap I \times \{\checkmark\} \neq \varnothing$ (because of $\mathcal{A}' = \langle\ \Sigma,\ Q \uplus \{\checkmark\},\ I, F,\ \Delta \uplus \langle \Sigma, F, \star \to \uparrow, \checkmark \rangle\ \rangle$)

Transformation Into BUTA Using Overloops

Almost the same the the loops-based version, but:

1. We compute sets of *overloops* instead of *loops*
2. Symbols are not stored in the states (not needed): thus $\Sigma \times \mathbb{T} \times 2^{Q^2}$ becomes $\mathbb{T} \times 2^{Q^2}$ again.
3. Final states are

$$\{\, (\star, O) \in \mathit{States} \mid O \cap (I \times \{\checkmark\}) \neq \varnothing \,\}$$

Acceptance criterion: Final Loops & Overloops

A term t is accepted iff

- $\mathcal{U}^{\tau}(t) \cap I \times F \neq \varnothing$
- $\mathfrak{V}^{\tau}(t) \cap I \times \{\checkmark\} \neq \varnothing$ (because of $\mathcal{A}' = \langle\, \Sigma,\ Q \uplus \{\checkmark\},\ I, F,\ \Delta \uplus \langle \Sigma, F, \star \to \uparrow, \checkmark \rangle \,\rangle$)

Transformation Into BUTA in the Deterministic Case

Deterministic TWA: Definition

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

[a] We do not need the usual, stronger definition, where I is a singleton.

Theorem: Deterministic Upper-Bound

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

Proof Idea (Full Proof in Appendix)

Sets of overloops on a given subterm are functional (ie. right-unique). Each computed state stores the set of overloops on a given subterm; thus there are at most $|Q+1|^{|Q|}$ of them, as opposed to $2^{|Q|^2}$ in the general case.

Transformation Into BUTA in the Deterministic Case

Deterministic TWA: Definition

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

[a] We do not need the usual, stronger definition, where I is a singleton.

Theorem: Deterministic Upper-Bound

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

Proof Idea (Full Proof in Appendix)

Sets of overloops on a given subterm are functional (ie. right-unique). Each computed state stores the set of overloops on a given subterm; thus there are at most $|Q+1|^{|Q|}$ of them, as opposed to $2^{|Q|^2}$ in the general case.

Transformation Into BUTA in the Deterministic Case

Deterministic TWA: Definition

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

[a] We do not need the usual, stronger definition, where I is a singleton.

Theorem: Deterministic Upper-Bound

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

Proof Idea (Full Proof in Appendix)

Sets of overloops on a given subterm are functional (ie. right-unique). Each computed state stores the set of overloops on a given subterm; thus there are at most $|Q+1|^{|Q|}$ of them, as opposed to $2^{|Q|^2}$ in the general case.

Testing emptiness of a TWA is an ExpTime-complete problem.
Practical problems:

- XML Queries
- Satisfiability of some XPath fragments
- But also model-checking...

Standard approach: tranform into BUTA, then test emptiness. We propose another approach:

- An "over-approximation"; *may* detect emptiness
- Executes in polynomial time and space
- Very (surprisingly) accurate in our random tests

Over-Approximation of the Emptiness Problem, Using Overloops

0. **Input:** An *escaped* TWA $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$
1. **Initialise** $\mathcal{L}_0$, $\mathcal{L}_1$, $\mathcal{L}_\star$ 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** $\mathcal{L}_0$, $\mathcal{L}_1$, $\mathcal{L}_\star$ 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 $\mathcal{L}_0$ and $\mathcal{L}_1$.
4. **Output:** *Empty* if $\mathcal{L}_\star \cap (I \times \{\checkmark\}) = \varnothing$, else *Unknown*

Over-Approximation of the Emptiness Problem, Using Overloops

0. **Input:** An *escaped* TWA $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$
1. **Initialise** $\mathcal{L}_0$, $\mathcal{L}_1$, $\mathcal{L}_\star$ 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** $\mathcal{L}_0$, $\mathcal{L}_1$, $\mathcal{L}_\star$ 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 $\mathcal{L}_0$ and $\mathcal{L}_1$.
4. **Output:** *Empty* if $\mathcal{L}_\star \cap (I \times \{\checkmark\}) = \varnothing$, else *Unknown*

S the set of simple loops built on $\mathcal{L}_0$ and $\mathcal{L}_1$.

Set of Simple Loops Built on the Sons (From Overloops)

$$S = \left\{ (p, q_\theta) \;\middle|\; \begin{array}{l} \exists \theta \in \mathbb{S} : \\ \exists p_\theta \in Q \end{array} \;\middle|\; \begin{array}{l} \langle f, p, \tau \to \chi(\theta), p_\theta \rangle \in \Delta \\ \text{and } (p_\theta, q_\theta) \in \mathcal{L}_\theta \end{array} \right\}$$

- Approach can be made coarser or finer: from a variant with no type information (ie. $\mathcal{L} = \mathcal{L}_{\boldsymbol{0}} \cup \mathcal{L}_{\boldsymbol{1}} \cup \mathcal{L}_{\star}$) to someting equivalent to transformation to BUTA
- The presented variant is polynomial in time and space
- Astonishing accuracy in random tests: Out of $\approx 20\,000$ TWA ($2 \leqslant |Q| \leqslant 20$), 75% of which had empty languages, only two *Unknown* instead of *Empty*.

Caveat

Our generation scheme was simplistic. Trivial instances? More tests to be made using uniform generation scheme.

Conclusion and Perspectives

What We Have Seen

- Two TWA Membership algorithms (loops & overloops)
- Two transformations from TWA into BUTA (idem)
- Overloops-based BUTA have expected states
- Overloops-based BUTA smaller (2 to 100 times...)
- DTWA: Lower upper-bound on BUTA states with overloops
- Approximation: polynomial, accurate in random tests

What Is Left To Do

- Test approximation with uniform DTWA generation scheme
- Characterise classes of TWA using overloops (det. etc)
- Significant size reductions on TWA (optimise queries etc)

Some References

[Comon et al., 2007, Samuelides, 2007, Héam et al., 2009, Bojańczyk, 2008]

Bojańczyk, M. (2008).

Tree-Walking Automata.

LATA'08 (tutorial), LNCS, 5196.

Comon, H., Dauchet, M., Gilleron, R., Löding, C., Jacquemard, F., Lugiez, D., Tison, S., and Tommasi, M. (2007).

Tree automata techniques and applications.

Héam, P.-C., Nicaud, C., and Schmitz, S. (2009).

Random Generation of Deterministic Tree (Walking) Automata.

In *CIAA'09*, volume 5642 of *LNCS*, pages 115–124.

Samuelides, M. (2007).

Automates d'arbres à jetons.

PhD thesis, Université Paris-Diderot - Paris VII.

Transformation Into BUTA Using Overloops

0. **Input:** An *escaped* 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 $P = (\tau, \mathcal{U}_a^\tau[\mathcal{H}_a^{\tau *}])$
     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 $f(P_0, P_1) \to P$ to *Rules* and P to *States* where $P_0, P_1 \in States$ such that $P_0 = (\mathbf{0}, S_0)$ and $P_1 = (\mathbf{1}, S_1)$ and $P = (\tau, \mathcal{U}_f^\tau[(\mathcal{H}_f^\tau \cup S)^*])$, with S the set of simple loops built on the sons.
4. **Output:** A BUTA $\mathcal{B}$ equivalent to $\mathcal{A}$: $\mathcal{B} = \langle \Sigma, States, \{ (\star, O) \in States \mid O \cap (I \times \{\checkmark\}) \neq \varnothing \}, Rules \rangle$

Transformation Into BUTA Using Overloops

0. **Input:** An *escaped* 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 $P = (\tau, \mathcal{U}_a^\tau[\mathcal{H}_a^{\tau *}])$
     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 $f(P_0, P_1) \to P$ to *Rules* and P to *States* where $P_0, P_1 \in States$ such that $P_0 = (\mathbf{0}, S_0)$ and $P_1 = (\mathbf{1}, S_1)$ and $P = (\tau, \mathcal{U}_f^\tau[(\mathcal{H}_f^\tau \cup S)^*])$, **with S the set of simple loops built on the sons.**
4. **Output:** A BUTA $\mathcal{B}$ equivalent to $\mathcal{A}$: $\mathcal{B} = \langle \Sigma, States, \{ (\star, O) \in States \mid O \cap (I \times \{\checkmark\}) \neq \varnothing \}, Rules \rangle$

Transformation Into BUTA
Using Overloops

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

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

Set of Simple Loops Built on the Sons (From Overloops)

$$S = \left\{ (p, q_\theta) \;\middle|\; \begin{array}{l} \exists \theta \in \mathbb{S} : \\ \exists p_\theta \in Q \end{array} \text{ st. } \begin{array}{l} \langle f, p, \tau \to \chi(\theta), p_\theta \rangle \in \Delta \\ \text{and } (p_\theta, q_\theta) \in S_\theta \end{array} \right\}$$

No need to store the son's symbol anywhere.

Transformation Into BUTA in the Deterministic Case

Theorem: Deterministic Upper-Bound

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

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

A relation $R \subseteq Q^2$ is **functional** (or **right-unique**, or **a partial function**) if, for all $p, q, q' \in Q$, pRq and $pRq' \implies q = q'$.

There are $2^{|Q|^2}$ binary relations on Q, of which $|Q+1|^{|Q|}$ are partial functions, of which $|Q|^{|Q|}$ are total functions.

If a relation R is functional, then so is R^k, for any $k \in \mathbb{N}$.

[a] We do not need the usual, stronger definition, where I is a singleton.

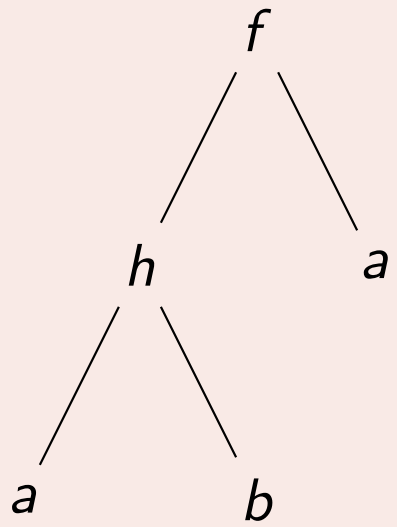
Transformation Into BUTA in the Deterministic Case

Idea: Each state built is the set of overloops (resp. loops) on some tree. We show that the set of overloops on a tree is functional.

Lemma: Deterministic TWA

If $\mathcal{A}$ is a deterministic TWA, then $\twoheadrightarrow_{\mathcal{A}}$ is functional.

Not enough to make sets of *loops* functional:



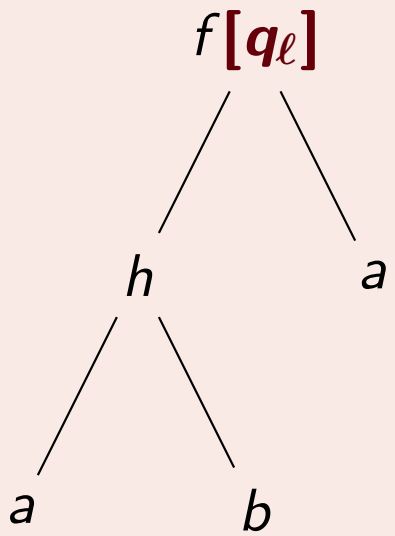
Transformation Into BUTA in the Deterministic Case

Idea: Each state built is the set of overloops (resp. loops) on some tree. We show that the set of overloops on a tree is functional.

Lemma: Deterministic TWA

If $\mathcal{A}$ is a deterministic TWA, then $\twoheadrightarrow_{\mathcal{A}}$ is functional.

Not enough to make sets of *loops* functional:



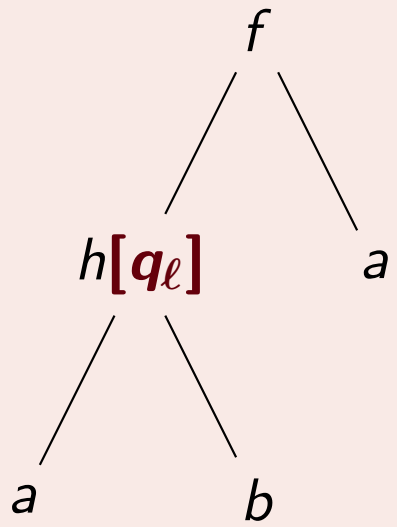
Transformation Into BUTA in the Deterministic Case

Idea: Each state built is the set of overloops (resp. loops) on some tree. We show that the set of overloops on a tree is functional.

Lemma: Deterministic TWA

If $\mathcal{A}$ is a deterministic TWA, then $\twoheadrightarrow_{\mathcal{A}}$ is functional.

Not enough to make sets of *loops* functional:



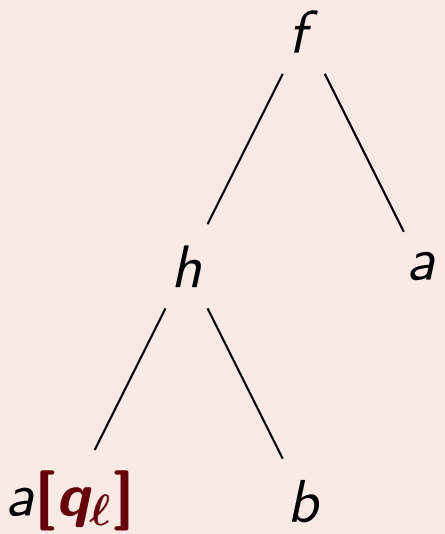
Transformation Into BUTA in the Deterministic Case

Idea: Each state built is the set of overloops (resp. loops) on some tree. We show that the set of overloops on a tree is functional.

Lemma: Deterministic TWA

If $\mathcal{A}$ is a deterministic TWA, then $\twoheadrightarrow_{\mathcal{A}}$ is functional.

Not enough to make sets of *loops* functional:



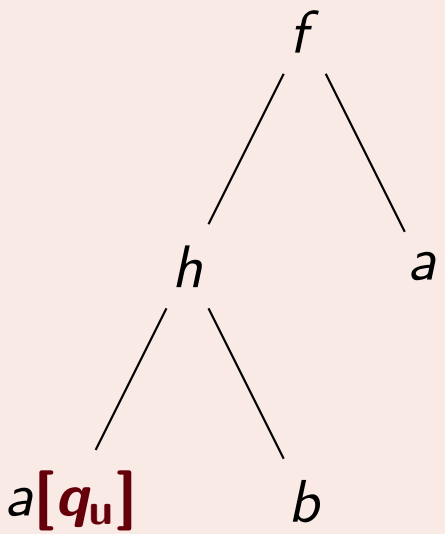
Transformation Into BUTA in the Deterministic Case

Idea: Each state built is the set of overloops (resp. loops) on some tree. We show that the set of overloops on a tree is functional.

Lemma: Deterministic TWA

If $\mathcal{A}$ is a deterministic TWA, then $\twoheadrightarrow_{\mathcal{A}}$ is functional.

Not enough to make sets of *loops* functional:



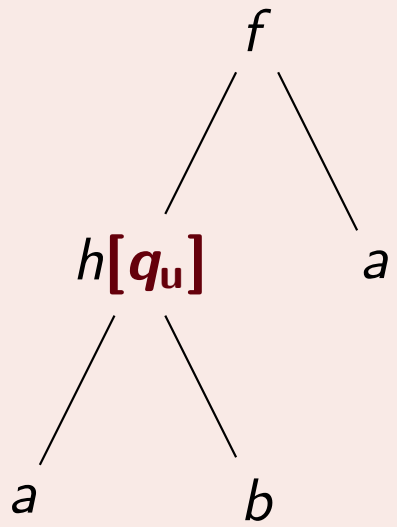
Transformation Into BUTA in the Deterministic Case

Idea: Each state built is the set of overloops (resp. loops) on some tree. We show that the set of overloops on a tree is functional.

Lemma: Deterministic TWA

If $\mathcal{A}$ is a deterministic TWA, then $\twoheadrightarrow_{\mathcal{A}}$ is functional.

Not enough to make sets of *loops* functional:



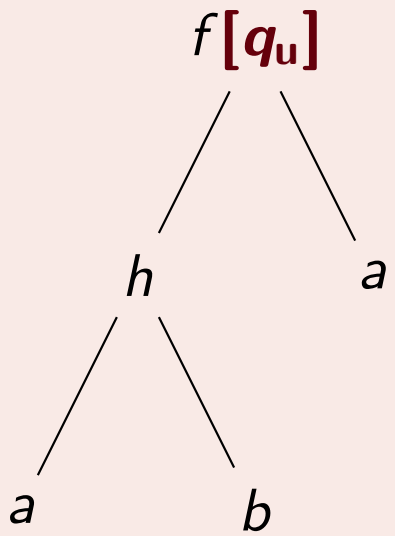
Transformation Into BUTA in the Deterministic Case

Idea: Each state built is the set of overloops (resp. loops) on some tree. We show that the set of overloops on a tree is functional.

Lemma: Deterministic TWA

If $\mathcal{A}$ is a deterministic TWA, then $\twoheadrightarrow_{\mathcal{A}}$ is functional.

Not enough to make sets of *loops* functional:



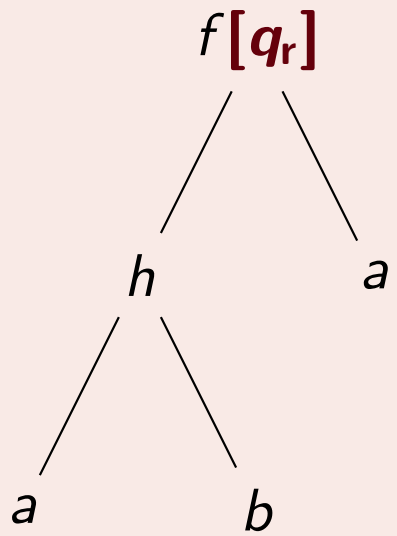
Transformation Into BUTA in the Deterministic Case

Idea: Each state built is the set of overloops (resp. loops) on some tree. We show that the set of overloops on a tree is functional.

Lemma: Deterministic TWA

If $\mathcal{A}$ is a deterministic TWA, then $\twoheadrightarrow_{\mathcal{A}}$ is functional.

Not enough to make sets of *loops* functional:



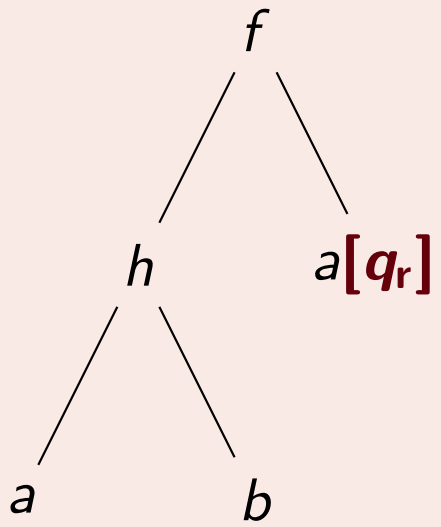
Transformation Into BUTA in the Deterministic Case

Idea: Each state built is the set of overloops (resp. loops) on some tree. We show that the set of overloops on a tree is functional.

Lemma: Deterministic TWA

If $\mathcal{A}$ is a deterministic TWA, then $\twoheadrightarrow_{\mathcal{A}}$ is functional.

Not enough to make sets of *loops* functional:



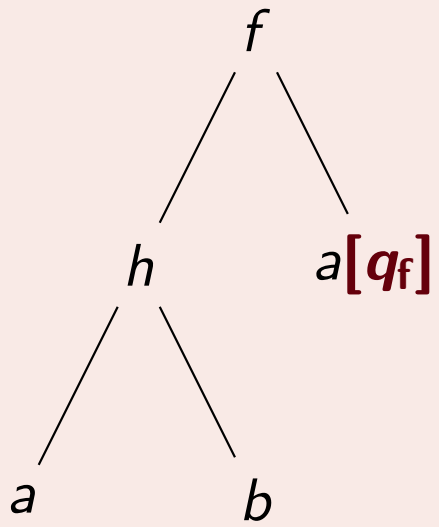
Transformation Into BUTA in the Deterministic Case

Idea: Each state built is the set of overloops (resp. loops) on some tree. We show that the set of overloops on a tree is functional.

Lemma: Deterministic TWA

If $\mathcal{A}$ is a deterministic TWA, then $\twoheadrightarrow_{\mathcal{A}}$ is functional.

Not enough to make sets of *loops* functional:



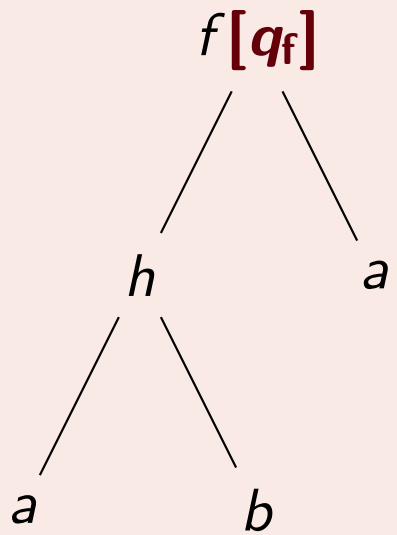
Transformation Into BUTA in the Deterministic Case

Idea: Each state built is the set of overloops (resp. loops) on some tree. We show that the set of overloops on a tree is functional.

Lemma: Deterministic TWA

If $\mathcal{A}$ is a deterministic TWA, then $\twoheadrightarrow_{\mathcal{A}}$ is functional.

Not enough to make sets of *loops* functional:



Transformation Into BUTA in the Deterministic Case

Idea: Each state built is the set of overloops (resp. loops) on some tree. We show that the set of overloops on a tree is functional.

Lemma: Deterministic TWA

If $\mathcal{A}$ is a deterministic TWA, then $\twoheadrightarrow_{\mathcal{A}}$ is functional.

Not enough to make sets of *loops* functional:
Loops on this tree: $\{(q_\ell, q_\ell), (q_\ell, q_u), (q_\ell, q_f), \dots\}$. Not functional.

Transformation Into BUTA in the Deterministic Case

Lemma: Hidden Loops

Let $p, q, q' \in Q$, $q \neq q'$ such that (p, q) and (p, q') are loops of a TWA $\mathcal{A}$ on a given subtree $t|_\alpha$. Then if $\mathcal{A}$ is deterministic, either (q, q') or (q', q) must be a loop of $\mathcal{A}$ on $t|_\alpha$.

By definition, there exist two runs $c_0, \ldots, c_n$ and $d_0, \ldots, d_m$ such that $c_0 = d_0 = (\alpha, p)$, $c_n = (\alpha, q)$ and $d_m = (\alpha, q')$.

- If $n = m$ then $c_0 \twoheadrightarrow^n c_n$ and $c_0 \twoheadrightarrow^n d_n$. It follows that $c_n = d_m$. But this contradicts $q \neq q'$, so we must have $n \neq m$.
- Say that $n < m$. Then $c_n = d_n$, and $(\alpha, q) = d_n, \ldots, d_m = (\alpha, q')$ forms a run. Therefore (q, q') is a loop.
- If $n > m$, then by the same arguments (q', q) is a loop.

Transformation Into BUTA in the Deterministic Case

Lemma: Hidden Loops

Let $p, q, q' \in Q$, $q \neq q'$ such that (p, q) and (p, q') are loops of a TWA $\mathcal{A}$ on a given subtree $t|_\alpha$. Then if $\mathcal{A}$ is deterministic, either (q, q') or (q', q) must be a loop of $\mathcal{A}$ on $t|_\alpha$.

By definition, there exist two runs $c_0, \ldots, c_n$ and $d_0, \ldots, d_m$ such that $c_0 = d_0 = (\alpha, p)$, $c_n = (\alpha, q)$ and $d_m = (\alpha, q')$.

- If $n = m$ then $c_0 \twoheadrightarrow^n c_n$ and $c_0 \twoheadrightarrow^n d_n$. It follows that $c_n = d_m$. But this contradicts $q \neq q'$, so we must have $n \neq m$.
- Say that $n < m$. Then $c_n = d_n$, and $(\alpha, q) = d_n, \ldots, d_m = (\alpha, q')$ forms a run. Therefore (q, q') is a loop.
- If $n > m$, then by the same arguments (q', q) is a loop.

Transformation Into BUTA in the Deterministic Case

Lemma: Functional Overloops

Let $p, q, q' \in Q$, such that (p, q) and (p, q') are overloops of a TWA $\mathcal{A}$ on a given subtree $t|_\alpha$. Then if $\mathcal{A}$ is deterministic, $q = q'$.

We have two runs

- $(\alpha, p), \ldots, (\alpha, s), (\overline{\mathfrak{p}}(\alpha), q)$
- $(\alpha, p), \ldots, (\alpha, s'), (\overline{\mathfrak{p}}(\alpha), q')$

Thus (p, s) and (p, s') are loops.

- If $s = s'$, then $q = q'$, because $\twoheadrightarrow$ is functional.
- If $s \neq s'$, then say (s, s') is a loop. So there exist $s_1, \ldots, s_n \in Q, \beta_1 \trianglelefteq \alpha, \ldots, \beta_n \trianglelefteq \alpha$ such that $(\alpha, s), (\beta_1, s_1), \ldots, (\beta_n, s_n), (\alpha, s')$ is a run. Thus $(\alpha, s) \twoheadrightarrow (\overline{\mathfrak{p}}(\alpha), q)$ and $(\alpha, s) \twoheadrightarrow (\beta_1, s_1)$. It follows that $\overline{\mathfrak{p}}(\alpha) = \beta_1 \trianglelefteq \alpha$: contradiction.

Transformation Into BUTA in the Deterministic Case

Lemma: Functional Overloops

Let $p, q, q' \in Q$, such that (p, q) and (p, q') are overloops of a TWA $\mathcal{A}$ on a given subtree $t|_\alpha$. Then if $\mathcal{A}$ is deterministic, $q = q'$.

We have two runs

- $(\alpha, p), \ldots, (\alpha, s), (\overline{\mathfrak{p}}(\alpha), q)$
- $(\alpha, p), \ldots, (\alpha, s'), (\overline{\mathfrak{p}}(\alpha), q')$

Thus (p, s) and (p, s') are loops.

- If $s = s'$, then $q = q'$, because $\twoheadrightarrow$ is functional.
- If $s \neq s'$, then say (s, s') is a loop. So there exist $s_1, \ldots, s_n \in Q, \beta_1 \trianglelefteq \alpha, \ldots, \beta_n \trianglelefteq \alpha$ such that $(\alpha, s), (\beta_1, s_1), \ldots, (\beta_n, s_n), (\alpha, s')$ is a run. Thus $(\alpha, s) \twoheadrightarrow (\overline{\mathfrak{p}}(\alpha), q)$ and $(\alpha, s) \twoheadrightarrow (\beta_1, s_1)$. It follows that $\overline{\mathfrak{p}}(\alpha) = \beta_1 \trianglelefteq \alpha$: contradiction.

Transformation Into BUTA in the Deterministic Case

Theorem: Deterministic Upper-Bound

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

By construction, for every state $P = (\tau, L)$ generated for $\mathcal{B}$ by the overloops-based algorithm, there exists at least a subtree t such that L is the set of overloops of $\mathcal{A}$ on t. Thus, by the previous lemma, L is functional. Therefore, there are at most $|\mathbb{T}| \cdot |Q+1|^{|Q|}$ states (or, equivalently, $|\mathbb{T}| \cdot 2^{|Q| \log_2(|Q|+1)}$).

Transformation Into BUTA in the Deterministic Case

Theorem: Deterministic Upper-Bound

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

By construction, for every state $P = (\tau, L)$ generated for $\mathcal{B}$ by the overloops-based algorithm, there exists at least a subtree t such that L is the set of overloops of $\mathcal{A}$ on t. Thus, by the previous lemma, L is functional. Therefore, there are at most $|\mathbb{T}| \cdot |Q+1|^{|Q|}$ states (or, equivalently, $|\mathbb{T}| \cdot 2^{|Q| \log_2(|Q|+1)}$).