Reachability Questions on (Partially Lossy) Queue Automata

28. Theorietag “Automaten und Formale Sprachen”, Wittenberg

Chris Köcher

Automata and Logics Group
Technische Universität Ilmenau

September 26, 2018
Let $A$ be an alphabet ($|A| \geq 2$).

Two actions for each $a \in A$:
- write letter $a \mapsto a$
- read letter $a \mapsto \bar{a}$

$\bar{A} := \{\bar{a} \mid a \in A\}$

$\Sigma := A \cup \bar{A}$
Queue Automata

Let $A$ be an alphabet ($|A| \geq 2$).

Two actions for each $a \in A$:
- write letter $a \rightsquigarrow a$
- read letter $a \rightsquigarrow \overline{a}$

$\overline{A} := \{\overline{a} \mid a \in A\}$

$\Sigma := A \cup \overline{A}$
Let $A$ be an alphabet ($|A| \geq 2$).

- Two actions for each $a \in A$:
  - write letter $a \rightsquigarrow a$
  - read letter $a \rightsquigarrow \bar{a}$

- $\overline{A} := \{\overline{a} \mid a \in A\}$
- $\Sigma := A \cup \overline{A}$
Queue Automata

- Let $A$ be an alphabet ($|A| \geq 2$).
- Two actions for each $a \in A$:
  - write letter $a \leadsto a$
  - read letter $a \leadsto \bar{a}$
- $\bar{A} := \{\bar{a} | a \in A\}$
- $\Sigma := A \cup \bar{A}$
Let $A$ be an alphabet ($|A| \geq 2$).

Two actions for each $a \in A$:
- write letter $a \rightsquigarrow a$
- read letter $a \rightsquigarrow \overline{a}$

$\overline{A} := \{\overline{a} \mid a \in A\}$

$\Sigma := A \cup \overline{A}$
Queue Automata

- Let $A$ be an alphabet ($|A| \geq 2$).
- Two actions for each $a \in A$:
  - write letter $a \mapsto a$
  - read letter $a \mapsto \bar{a}$
- $\bar{A} := \{\bar{a} \mid a \in A\}$
- $\Sigma := A \cup \bar{A}$
Let $A$ be an alphabet ($|A| \geq 2$).

Two actions for each $a \in A$:
- write letter $a \mapsto a$
- read letter $a \mapsto \overline{a}$

- $\overline{A} := \{\overline{a} | a \in A\}$
- $\Sigma := A \cup \overline{A}$
Let $A$ be an alphabet ($|A| \geq 2$).

Two actions for each $a \in A$:

- write letter $a \mapsto a$
- read letter $a \mapsto \overline{a}$

- $\overline{A} := \{\overline{a} \mid a \in A\}$
- $\Sigma := A \cup \overline{A}$
Reachability Problems

Inputs:
- \( T \subseteq \Sigma^* \) regular language of transformations
- \( L \subseteq A^* \) regular language of queue contents

Compute:
- \( \text{post}_T(L) := \left\{ q \in A^* \mid \exists p \in L, t \in T: p \xrightarrow{t} q \right\} \)
- \( \text{pre}_T(L) := \left\{ q \in A^* \mid \exists p \in L, t \in T: q \xrightarrow{t} p \right\} \)
Reachability Problems

Inputs:
- $T \subseteq \Sigma^*$ regular language of transformations
- $L \subseteq A^*$ regular language of queue contents

Compute:
- $\text{post}_T(L) := \{q \in A^* \mid \exists p \in L, t \in T: p \xrightarrow{t} q\}$
- $\text{pre}_T(L) := \{q \in A^* \mid \exists p \in L, t \in T: q \xrightarrow{t} p\}$

Example
Known Results

Theorem (Brand, Zafiropulo 1983)
Queue Automata are Turing-complete.

- $\text{post}_T(\varepsilon) / \text{pre}_T(\varepsilon)$ can be any recursively enumerable language
- holds already for $T = \{w_1, \ldots, w_n\}^*$ with $w_1, \ldots, w_n \in \Sigma^*$

Theorem (Boigelot, Godefroid, Willems, Wolper 1997)
For $w \in \Sigma^*$ and $L \subseteq A^*$ regular, the sets $\text{post}_{w^*}(L)$ and $\text{pre}_{w^*}(L)$ are effectively regular.
Behavioral Equivalence: Definition

Definition

\( s, t \in \Sigma^* \) behave equally (in symbols \( s \equiv t \)) if, and only if,

\[
\forall p, q \in A^*: p \xrightarrow{s} q \iff p \xrightarrow{t} q
\]

\[ \Rightarrow \text{post}_s(L) = \text{post}_t(L) \text{ and pre}_s(L) = \text{pre}_t(L) \]

Theorem (Huschenbett, Kuske, Zetzsche 2014)

\( \equiv \) is the least congruence on \( \Sigma^* \) satisfying the following equations:

1. \( ab \equiv ba \) if \( a \neq b \)
2. \( aac \equiv aac \)
3. \( ca\bar{a} \equiv c\bar{a}a \)

for any \( a, b, c \in A \).
Corollary

Let $T \subseteq \Sigma^*$ be closed under $\equiv$ and $L \subseteq A^*$. Then
\[
\text{post}_T(L) = \text{post}_{T \cap A^* A^* A^*}(L) \quad \text{and} \quad \text{pre}_T(L) = \text{pre}_{T \cap A^* A^* A^*}(L).
\]

Theorem

Let $T \subseteq \Sigma^*$ be regular and closed under $\equiv$ and $L \subseteq A^*$ be regular. Then $\text{post}_T(L)$ and $\text{pre}_T(L)$ are effectively regular (in polynomial time).

Proof. We have $T \cap A^* A^* A^* = \bigcup_i K_{i,1} K_{i,2} K_{i,3}$ for finitely many regular languages $K_{i,1}, K_{i,2}, K_{i,3} \subseteq A^*$. Then
\[
\text{post}_T(L) = \text{post} \bigcup_i K_{i,1} K_{i,2} K_{i,3}(L) = \bigcup_i K_{i,3} \setminus ((K_{i,1} \setminus L) \cdot K_{i,2})
\]
which is regular since class of regular languages is closed under quotients and products.
Behavioral Equivalence: Result

Corollary

Let $T \subseteq \Sigma^*$ be closed under $\equiv$ and $L \subseteq A^*$. Then

$$\text{post}_T(L) = \text{post}_{T \cap \overline{A^*}} \overline{A^*}(L) \quad \text{and} \quad \text{pre}_T(L) = \text{pre}_{T \cap \overline{A^*}} \overline{A^*}(L).$$

Theorem

Let $T \subseteq \Sigma^*$ be regular and closed under $\equiv$ and $L \subseteq A^*$ be regular. Then $\text{post}_T(L)$ and $\text{pre}_T(L)$ are effectively regular (in polynomial time).

Example

Let $R_1, R_2 \subseteq A^*$ be regular. $R_1 \sqcup \overline{R_2}$ is regular and closed under $\equiv$. 
Read-Write Independence: Definition

Definition

Let $T \subseteq \Sigma^*$. $T$ is read-write independent if for all $s, t \in T$ there is $u_{s,t} \in T$ with

- $\text{write}(u_{s,t}) = \text{write}(s)$ and
- $\text{read}(u_{s,t}) = \text{read}(t)$.

Example

- $\{w\}$ for $w \in \Sigma^*$
- $\overline{KLM}$ for $K, L, M \subseteq A^*$
- $\text{Perm}(w)$ ... the set of all permutations of $w \in \Sigma^*$
Read-Write Independence: Result

**Theorem**

Let $T \subseteq \Sigma^*$ be finite and read-write independent and $L \subseteq A^*$ be regular. Then $\text{post}_{T^*}(L)$ and $\text{pre}_{T^*}(L)$ are effectively regular (in polynomial space).

**Conjecture**

$\text{post}_{T^*}(L)$ and $\text{pre}_{T^*}(L)$ are effectively regular even if

- $T$ is regular, closed under $\equiv$, and read-write independent.
- $T = \overline{K} \overline{L} \overline{M}$ for $K, L, M \subseteq A^*$ regular.
Read-Write Independence: Proof - Overview

$\in L$

$t_1$

$t_2$

$t_3$

$t_4$

$t_5$

$t_6$
Read-Write Independence: Proof - Overview

Phase 1

\[ \in L \]

\( t_1 \)
\( t_2 \)
\( t_3 \)

\( t_4 \)
\( t_5 \)
\( t_6 \)
Read-Write Independence: Proof - Phases 1 + 2

1. Read only letters from the queue’s initial contents

   - Let $\mathcal{A} = (Z, A, I, \delta, F)$ be NFA accepting $L$.
   - for each $z \in Z$:
     - Compute $R_z := L(\mathcal{A}_{F:={z}}) \cap \text{read}(T)^*$
     - Replace each occurrence of a word from $\text{read}(T)$ in $R_z$ by $\text{write}(T)$ \[\rightsquigarrow\] $W_z \subseteq \text{write}(T)^*$
     - Result: $L(\mathcal{A}_{I:={z}}) \cdot W_z$
   - $X_1 := \bigcup_{z \in Z} L(\mathcal{A}_{I:={z}}) \cdot W_z$
Read-Write Independence: Proof - Overview

Phase 1

Phase 2

Phase 3

$t_1 \in L$

$t_2$

$t_3$

$t_4$

$t_5$

$t_6$
1 Read only letters from the queue’s initial contents

- Let $A = (Z, A, I, \delta, F)$ be NFA accepting $L$.
- for each $z \in Z$:
  - Compute $R_z := L(A_{F:=\{z\}}) \cap \text{read}(T)^*$
  - Replace each occurrence of a word from $\text{read}(T)$ in $R_z$ by $\text{write}(T) \rightsquigarrow W_z \subseteq \text{write}(T)^*$
  - Result: $L(A_{I:=\{z\}}) \cdot W_z$
- $X_1 := \bigcup_{z \in Z} L(A_{I:=\{z\}}) \cdot W_z$

2 Read some letters from the queue’s initial contents and some letters written in Phase 1

- $X_2 := \text{post}_T(X_1) \cap \text{suffixes(write}(T)) \cdot \text{write}(T)^*$
Read-Write Independence: Proof - Overview

Phase 1

Phase 2

Phase 3

$\in L$

$t_1$

$t_2$

$t_3$

$t_4$

$t_5$

$t_6$

Phase 3
Read only letters written in Phases 1-3

\[ X_2 = Y_0 \xrightarrow{T} Y_1 \xrightarrow{T} \cdots \xrightarrow{T} Y_{m-1} \xrightarrow{T} Y_m, Y_n \]

- \( Y_i = \text{post}_{T^i}(X_2) = \bigcup_{s \in S_i} s \text{write}(T)^{E_{s,i}} \)
  - \( S_i \subseteq \text{suffixes(\text{write}(T))} \) minimal
  - \( E_{s,i} \subseteq \mathbb{N} \) semi-linear
- \( S_m = S_n \) and arithmetical difference of the \( E_{s,n} \)'s and \( E_{s,m} \)'s is effectively semi-linear
- \( X_3 \) is effectively semi-linear

\[ \Rightarrow \text{post}_{T^*}(L) = X_1 \cup X_2 \cup X_3. \]
can forget some contents of their queue content at any time

Known Results (for fully lossy queues):

- $\text{post}_T(L)$ is regular
- NFA accepting $\text{post}_T(L)$ cannot be computed [Mayr 2003]
- Membership of $\text{post}_T(L)$ is decidable [Abdulla, Jonsson 1996], but not primitive recursive [Schnoebelen 2002]
- $\text{pre}_T(L)$ is effectively regular [Abdulla, Jonsson 1996]

Results from this talk hold for arbitrary Partially Lossy Queue Automata

Thank you!