TECHNISCHE UNIVERSITÄT ILMENAU

Reachability Problems on (Partially Lossy) Queue Automata

13th International Conference on Reachability Problems, Brussels

Chris Köcher

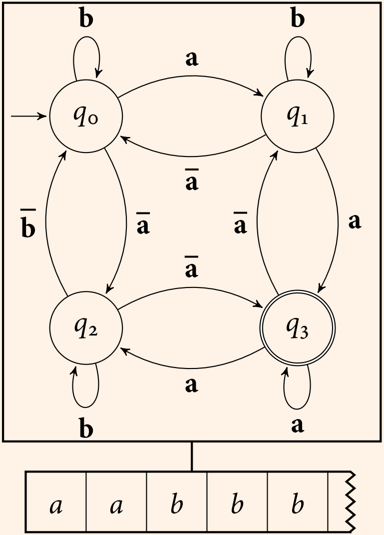
Automata and Logics Group
Technische Universität Ilmenau

September 11, 2019

Queue Automata

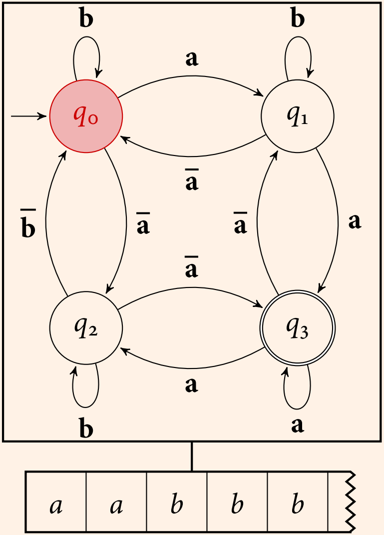
- Let A be an alphabet.
- Two actions for each $a \in A$:
  - write letter $a \rightsquigarrow \mathbf{a}$
  - read letter $a \rightsquigarrow \overline{\mathbf{a}}$
- $\mathbf{A} := \{\mathbf{a} \mid a \in A\}$, $\overline{\mathbf{A}} := \{\overline{\mathbf{a}} \mid a \in A\}$
- $\Sigma := \mathbf{A} \uplus \overline{\mathbf{A}}$

Example



- Let A be an alphabet.
- Two actions for each $a \in A$:
  - write letter $a \rightsquigarrow \mathbf{a}$
  - read letter $a \rightsquigarrow \overline{\mathbf{a}}$
- $\mathbf{A} := \{\mathbf{a} \mid a \in A\}$, $\overline{\mathbf{A}} := \{\overline{\mathbf{a}} \mid a \in A\}$
- $\Sigma := \mathbf{A} \uplus \overline{\mathbf{A}}$

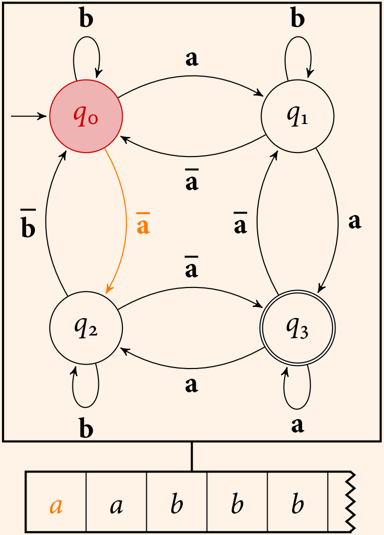
Example



Queue Automata

- Let A be an alphabet.
- Two actions for each $a \in A$:
  - write letter $a \rightsquigarrow \mathbf{a}$
  - read letter $a \rightsquigarrow \overline{\mathbf{a}}$
- $\mathbf{A} := \{\mathbf{a} \mid a \in A\}$, $\overline{\mathbf{A}} := \{\overline{\mathbf{a}} \mid a \in A\}$
- $\Sigma := \mathbf{A} \uplus \overline{\mathbf{A}}$

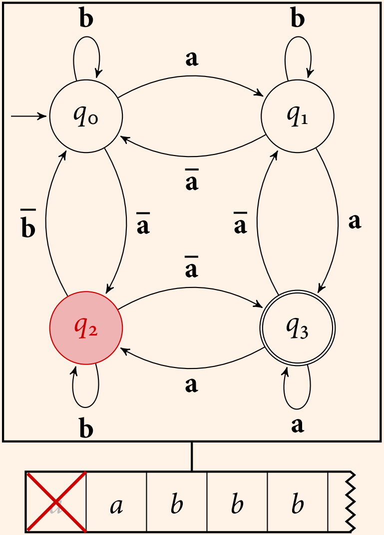
Example



Queue Automata

- Let A be an alphabet.
- Two actions for each $a \in A$:
  - write letter $a \rightsquigarrow \mathbf{a}$
  - read letter $a \rightsquigarrow \bar{\mathbf{a}}$
- $\mathbf{A} := \{\mathbf{a} \mid a \in A\}$, $\bar{\mathbf{A}} := \{\bar{\mathbf{a}} \mid a \in A\}$
- $\Sigma := \mathbf{A} \uplus \bar{\mathbf{A}}$

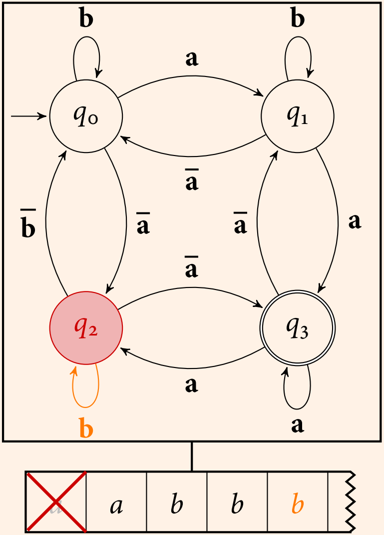
Example



Queue Automata

- Let A be an alphabet.
- Two actions for each $a \in A$:
  - write letter $a \rightsquigarrow \mathbf{a}$
  - read letter $a \rightsquigarrow \overline{\mathbf{a}}$
- $\mathbf{A} := \{\mathbf{a} \mid a \in A\}$, $\overline{\mathbf{A}} := \{\overline{\mathbf{a}} \mid a \in A\}$
- $\Sigma := \mathbf{A} \uplus \overline{\mathbf{A}}$

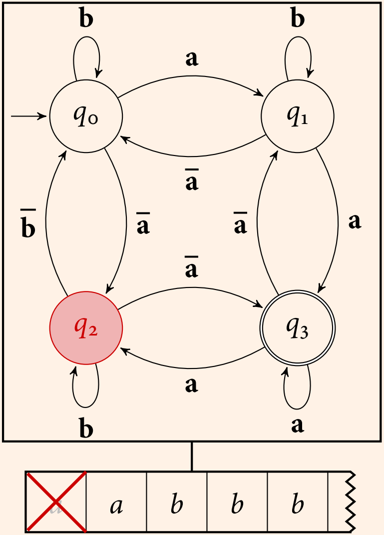
Example



Queue Automata

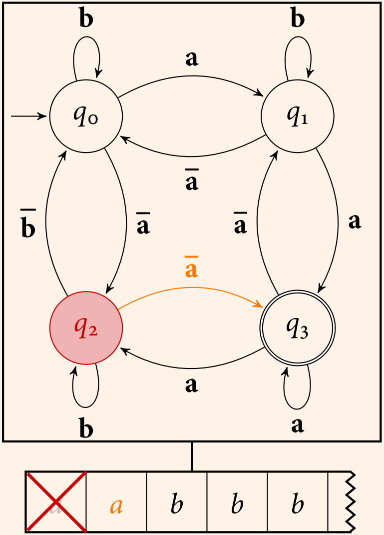
- Let A be an alphabet.
- Two actions for each $a \in A$:
  - write letter $a \rightsquigarrow \mathbf{a}$
  - read letter $a \rightsquigarrow \overline{\mathbf{a}}$
- $\mathbf{A} := \{\mathbf{a} \mid a \in A\}$, $\overline{\mathbf{A}} := \{\overline{\mathbf{a}} \mid a \in A\}$
- $\Sigma := \mathbf{A} \uplus \overline{\mathbf{A}}$

Example



- Let A be an alphabet.
- Two actions for each $a \in A$:
  - write letter $a \rightsquigarrow \mathbf{a}$
  - read letter $a \rightsquigarrow \overline{\mathbf{a}}$
- $\mathbf{A} := \{\mathbf{a} \mid a \in A\}$, $\overline{\mathbf{A}} := \{\overline{\mathbf{a}} \mid a \in A\}$
- $\Sigma := \mathbf{A} \uplus \overline{\mathbf{A}}$

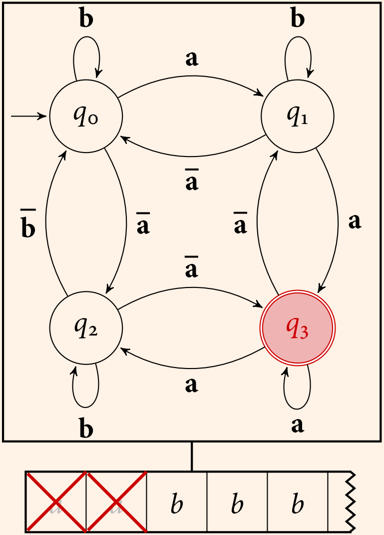
Example



Queue Automata

- Let A be an alphabet.
- Two actions for each $a \in A$:
  - write letter $a \rightsquigarrow \mathbf{a}$
  - read letter $a \rightsquigarrow \overline{\mathbf{a}}$
- $\mathbf{A} := \{\mathbf{a} \mid a \in A\}$, $\overline{\mathbf{A}} := \{\overline{\mathbf{a}} \mid a \in A\}$
- $\Sigma := \mathbf{A} \uplus \overline{\mathbf{A}}$

Example



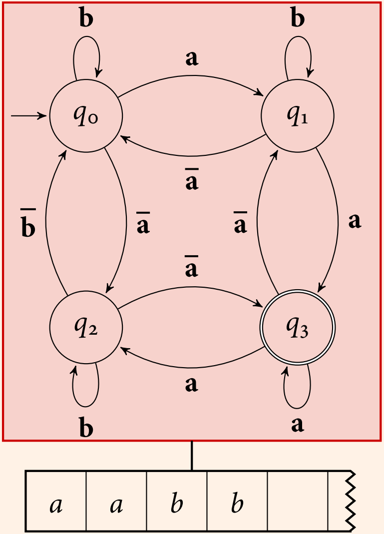
Inputs:

- $\mathbf{T} \subseteq \Sigma^*$ regular language of transformation sequences
- $L \subseteq A^*$ regular language of queue contents

Compute:

- $\textsc{Reach}(L, \mathbf{T})$:= the set of all queue contents after application of $\mathbf{T}$ on L

Example



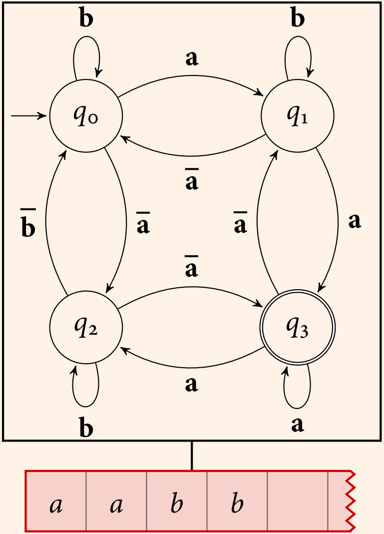
Inputs:

- $\mathbf{T} \subseteq \boldsymbol{\Sigma}^*$ regular language of transformation sequences
- $L \subseteq A^*$ regular language of queue contents

Compute:

- $\textsc{Reach}(L, \mathbf{T})$:= the set of all queue contents after application of $\mathbf{T}$ on L

Example



Turing-Completeness

Theorem (Brand, Zafiropulo 1983)

Queue Automata can simulate Turing-machines.

- $\textsc{Reach}(L, \mathbf{T})$ can be any recursively enumerable language
- holds already for some fixed $\mathbf{T} = \{\mathbf{t_1}, \ldots, \mathbf{t_n}\}^*$ with $\mathbf{t_1}, \ldots, \mathbf{t_n} \in \mathbf{\Sigma^*}$

Approximations of the Reachability Problem

TECHNISCHE UNIVERSITÄT ILMENAU

- Iterative approach: for $i = 0, 1, 2, \ldots$ do
  - compute the prefixes $\mathbf{T_i}$ of length i from $\mathbf{T}$
  - apply $\mathbf{T_i}$ on L
- Faster approach:

Theorem (Boigelot, Godefroid, Willems, Wolper 1997)

Let $L \subseteq A^*$ be regular and $\mathbf{t} \in \mathbf{\Sigma^*}$. Then $\textsc{Reach}(L, \mathbf{t^*})$ *is effectively regular.*

$\Rightarrow$ Combine multiple iterations of a loop to a meta-transformation

Aim

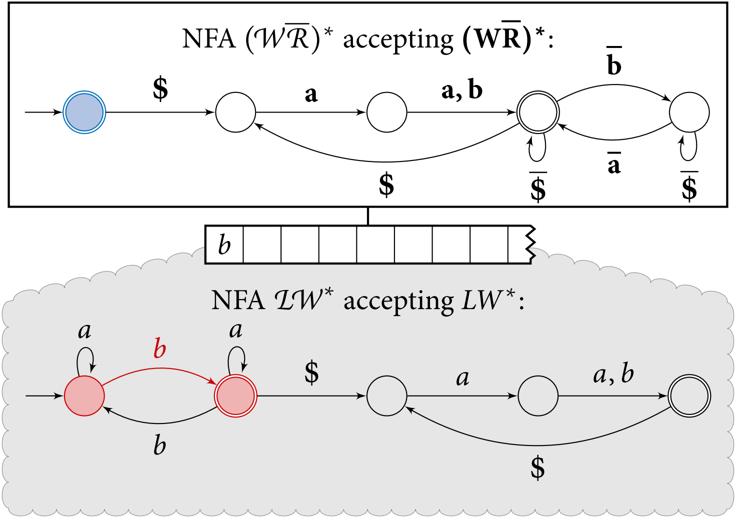
Generalize this result.

Theorem

Let $L, W, R \subseteq A^*$ be regular. Then $\textsc{Reach}(L, (\mathbf{W}\overline{\mathbf{R}})^*)$ *is effectively regular (in polynomial time).*

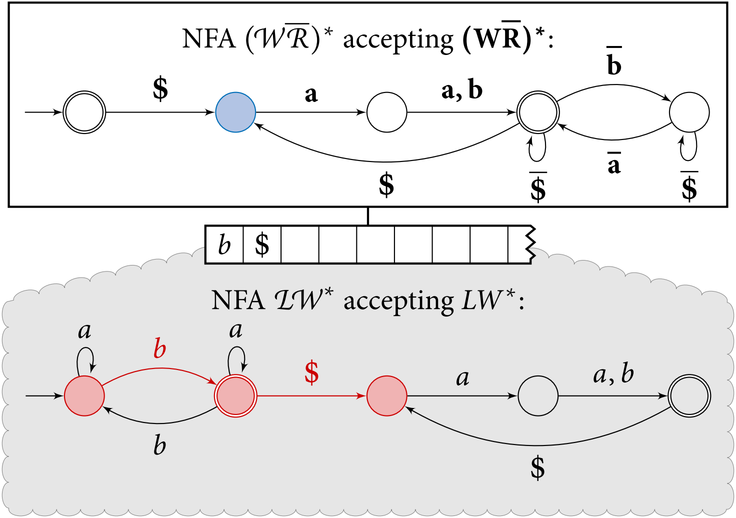
- We slightly modify W and R:
  - Let $\$ \notin A$ be some new letter.
  - Set $\mathbf{W'} := \mathbf{\$W}$ and $\overline{\mathbf{R'}} := \text{shuffle}(\overline{\mathbf{R}}, \overline{\mathbf{\$}}^*)$.
  - Easy: $\textsc{Reach}(L, (\mathbf{W}\overline{\mathbf{R}})^*) = \text{proj}_A(\textsc{Reach}(L, (\mathbf{W'}\overline{\mathbf{R'}})^*))$.
  - We prove that $\textsc{Reach}(L, (\mathbf{W'}\overline{\mathbf{R'}})^*)$ is regular.
- From now on, we write W and $\overline{R}$ instead of W' and $\overline{R'}$, resp.

- Consider the following example:



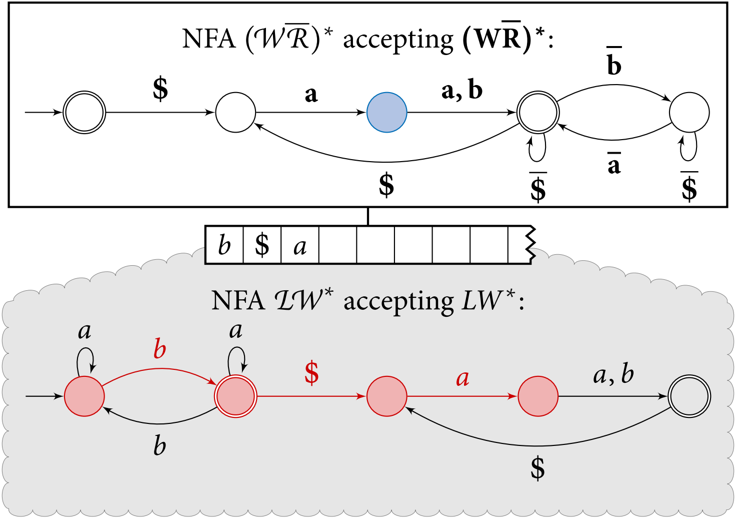
Proof Idea (1)

- Consider the following example:

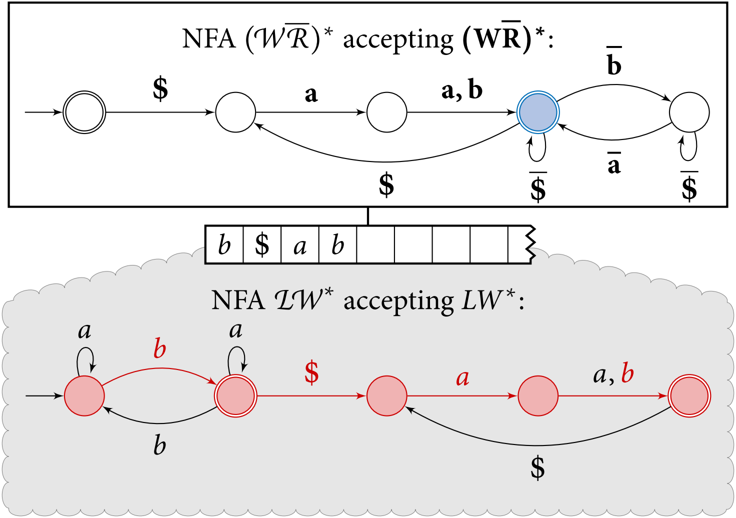


Proof Idea (1)

- Consider the following example:

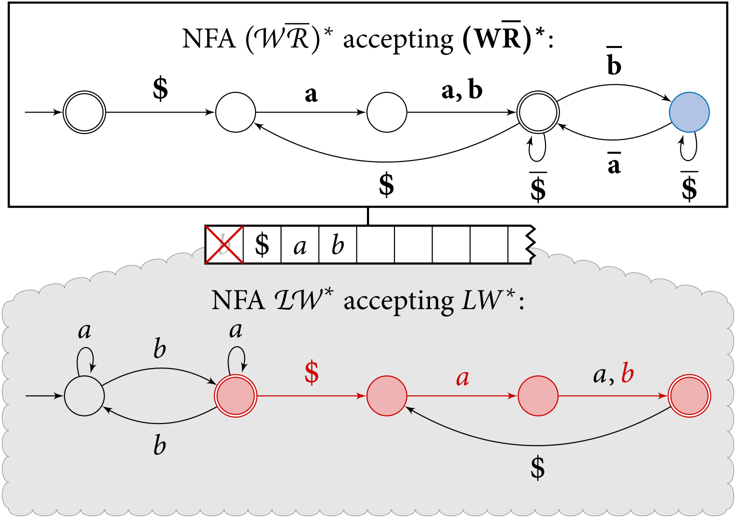


- Consider the following example:



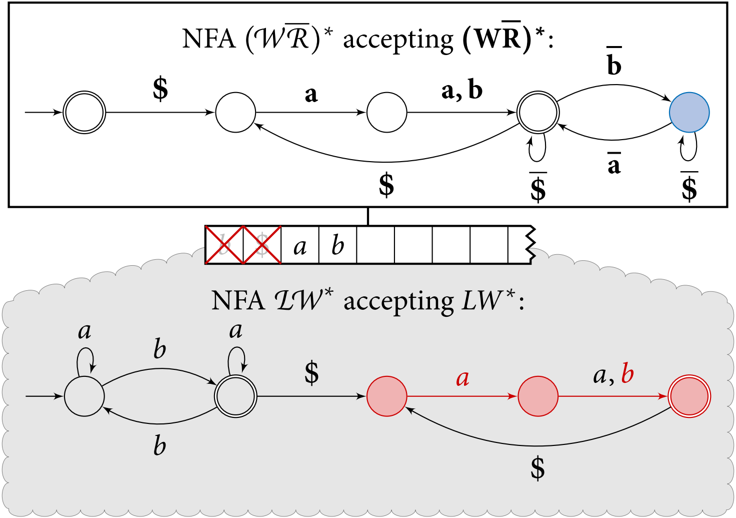
TECHNISCHE UNIVERSITÄT ILMENAU

- Consider the following example:



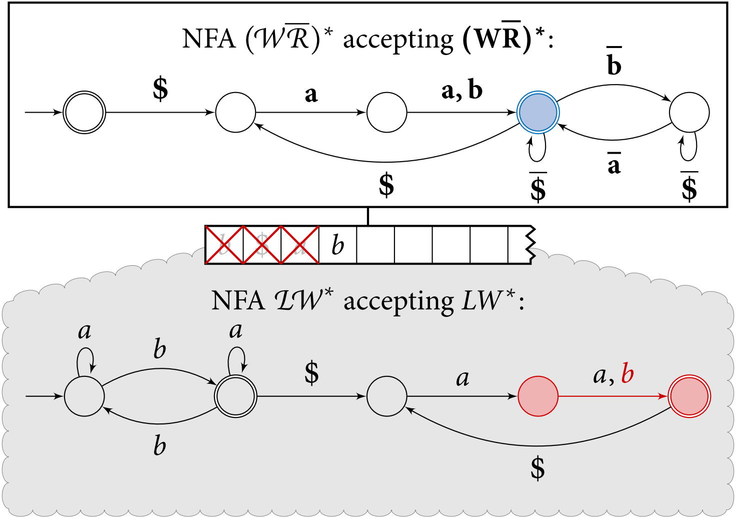
Proof Idea (1)

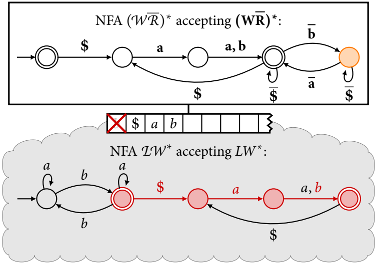
- Consider the following example:



Proof Idea (1)

- Consider the following example:





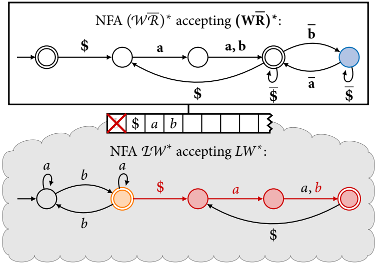
- A configuration of the queue automaton can be abstracted as follows:
  1. the current state in $(\mathcal{W}\overline{\mathcal{R}})^*$
  2. the starting state of the path in $\mathcal{LW}^*$
  3. the ending state of the path in $\mathcal{LW}^*$
  4. the number of \$s on the path

  (items 1–3: control state of $\mathcal{C}$; item 4: counter of $\mathcal{C}$)

⇒ The queue automaton can be simulated by a one-counter automaton $\mathcal{C}$

Proof Idea (2)

TECHNISCHE UNIVERSITÄT ILMENAU

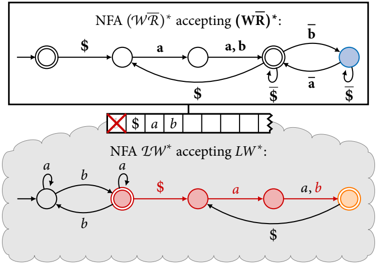


- A configuration of the queue automaton can be abstracted as follows:
  1. the current state in $(\mathcal{W}\overline{\mathcal{R}})^*$
  2. the starting state of the path in $\mathcal{LW}^*$
  3. the ending state of the path in $\mathcal{LW}^*$
  4. the number of \$s on the path

  (1–3: control state of $\mathcal{C}$; 4: counter of $\mathcal{C}$)

⇒ The queue automaton can be simulated by a one-counter automaton $\mathcal{C}$

Proof Idea (2)

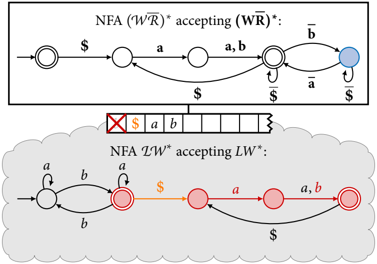


- A configuration of the queue automaton can be abstracted as follows:
  1. the current state in $(\mathcal{W}\overline{\mathcal{R}})^*$
  2. the starting state of the path in $\mathcal{LW}^*$
  3. the ending state of the path in $\mathcal{LW}^*$
  4. the number of \$s on the path

  (1–3) } control state of C
  (4) } counter of C

$\Rightarrow$ The queue automaton can be simulated by a one-counter automaton C

Proof Idea (2)



- A configuration of the queue automaton can be abstracted as follows:
  1. the current state in $(\mathcal{W}\overline{\mathcal{R}})^*$
  2. the starting state of the path in $\mathcal{LW}^*$
  3. the ending state of the path in $\mathcal{LW}^*$

  (1–3: control state of $\mathcal{C}$)

  4. the number of \$s on the path

  (4: counter of $\mathcal{C}$)

$\Rightarrow$ The queue automaton can be simulated by a one-counter automaton $\mathcal{C}$

Semantics of $\mathcal{C}$

- $\mathcal{C}$'s configurations consist of:
  1. the current state in $(\mathcal{W}\overline{\mathcal{R}})^*$
  2. the starting state of the path in $\mathcal{LW}^*$
  3. the ending state of the path in $\mathcal{LW}^*$
  4. the number of \$s on the path

  (1–3: control state of $\mathcal{C}$; 4: counter of $\mathcal{C}$)
- Let $(p, q, r, n) \in \mathrm{Conf}_{\mathcal{C}}$ be a configuration of $\mathcal{C}$.
- $[\![p, q, r, n]\!] := L(\mathcal{LW}^*_{q \to r}) \cap \mathrm{shuffle}(\$^n, A^*)$

Proposition

$$\textsc{Reach}(L, (\mathbf{W}\overline{\mathbf{R}})^*) = \bigcup_{\sigma \in \mathrm{Conf}_{\mathcal{C}},\ \textit{reach.} + \textit{acc.}} [\![\sigma]\!]\,,$$

i.e., $\textsc{Reach}(L, (\mathbf{W}\overline{\mathbf{R}})^*)$ *is a rational image of the set of reachable and accepting configurations of* $\mathcal{C}$.

Finishing the Proof

- Consider the set of reachable and accepting configurations of C.
- By [Bouajjani, Esparza, Maler 1997] this set is semilinear.
- Using a rational transduction implies effective regularity of $\textsc{Reach}(L, (\mathbf{W\overline{R}})^*)$. □

⇒ We have seen:

Theorem (Main Theorem)

Let $L, W, R \subseteq A^*$ be regular. Then $\textsc{Reach}(L, (\mathbf{W\overline{R}})^*)$ is effectively regular (in polynomial time).

Consequences

TECHNISCHE UNIVERSITÄT ILMENAU

Corollary

Let $L \subseteq A^*$ and $\mathbf{T} \subseteq \mathbf{\Sigma}^*$ be regular. Then $\textsc{Reach}(L, \mathbf{T}^*)$ is regular if

1. $\mathbf{T} = \overline{\mathbf{R_1}}\mathbf{W}\overline{\mathbf{R_2}}$ *for regular* $W, R_1, R_2 \subseteq A^*$,
2. $\mathbf{T} = \mathbf{W} \cup \overline{\mathbf{R}}$ *for regular* $W, R \subseteq A^*$,
3. $\mathbf{T} = \{\mathbf{t}\}$ *for* $\mathbf{t} \in \mathbf{\Sigma}^*$ *(cf. [Boigelot et al. 1997]), or*
4. $\mathbf{T} = \text{shuffle}(\mathbf{W}, \overline{\mathbf{R}})$ *for regular* $W, R \subseteq A^*$.

- Remark: Proofs of 3 and 4 use some result from [K. 2018, cf. STACS'18]

Thank you!