Complexity of checking whether two automata are synchronized by the same language

Marina Maslennikova

RuFiDiM 2014
Petrozavodsk, Russia, September 15–18, 2014

Institute of Mathematics and Computer Science
Ural Federal University, Ekaterinburg, Russia

Definitions

- A *deterministic finite automaton* (DFA) is a triple $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$, where
  – Q is the state set;
  – Σ is the input alphabet;
  – $\delta : Q \times \Sigma \to Q$ is totally defined transition function.
  We do not need any initial and final states.
- Σ^* stands for the set of all words over Σ including the empty word.
- The function δ uniquely extends to a function $Q \times \Sigma^* \to Q$ still denoted by δ.
- We often write $q \,.\, w$ for $\delta(q, w)$ and $P \,.\, w$ for $\delta(P, w) = \{\delta(q, w) \mid q \in P\}$.

- A *deterministic finite automaton* (DFA) is a triple $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$, where
  – Q is the state set;
  – Σ is the input alphabet;
  – $\delta : Q \times \Sigma \to Q$ is totally defined transition function.
  We do not need any initial and final states.
- Σ^* stands for the set of all words over Σ including the empty word.
- The function δ uniquely extends to a function $Q \times \Sigma^* \to Q$ still denoted by δ.
- We often write $q \,.\, w$ for $\delta(q, w)$ and $P \,.\, w$ for $\delta(P, w) = \{\delta(q, w) \mid q \in P\}$.

Definitions

- A *deterministic finite automaton* (DFA) is a triple $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$, where
  – Q is the state set;
  – Σ is the input alphabet;
  – $\delta : Q \times \Sigma \to Q$ is totally defined transition function.
  We do not need any initial and final states.
- Σ^* stands for the set of all words over Σ including the empty word.
- The function δ uniquely extends to a function $Q \times \Sigma^* \to Q$ still denoted by δ.
- We often write $q \,.\, w$ for $\delta(q, w)$ and $P \,.\, w$ for $\delta(P, w) = \{\delta(q, w) \mid q \in P\}$.

Definitions

- A *deterministic finite automaton* (DFA) is a triple $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$, where
  – Q is the state set;
  – Σ is the input alphabet;
  – $\delta : Q \times \Sigma \to Q$ is totally defined transition function.
  We do not need any initial and final states.
- Σ^* stands for the set of all words over Σ including the empty word.
- The function δ uniquely extends to a function $Q \times \Sigma^* \to Q$ still denoted by δ.
- We often write $q \,.\, w$ for $\delta(q, w)$ and $P \,.\, w$ for $\delta(P, w) = \{\delta(q, w) \mid q \in P\}$.

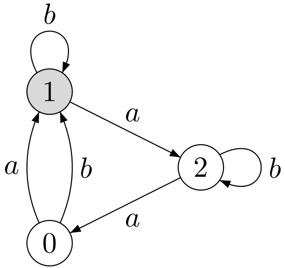
Definitions

- A DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ is called *synchronizing* if there exists a word $w \in \Sigma^*$ whose action resets $\mathscr{A}$, that is $|Q \, . \, w| = 1$.
- Any word with this property is said to be *reset* for the DFA $\mathscr{A}$.
- $\mathrm{Syn}(\mathscr{A})$ is the language of all reset words for $\mathscr{A}$.

Definitions

- A DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ is called *synchronizing* if there exists a word $w \in \Sigma^*$ whose action resets $\mathscr{A}$, that is $|Q \,.\, w| = 1$.
- Any word with this property is said to be *reset* for the DFA $\mathscr{A}$.
- $\mathrm{Syn}(\mathscr{A})$ is the language of all reset words for $\mathscr{A}$.

Example



A reset word is $w = baab$. Indeed, $Q\,.\,w = \{1\}$. In fact this is the shortest reset word for this automaton.

- A language $I \subseteq \Sigma^*$ is called a *two-sided ideal* (or simply an *ideal*) if $I \neq \emptyset$ and $I = \Sigma^* I \Sigma^*$.
- For every synchronizing automaton $\mathscr{A}$ it holds

$$\Sigma^* \operatorname{Syn}(\mathscr{A})\Sigma^* = \operatorname{Syn}(\mathscr{A}).$$

Ideals and Synchronizing Automata

- A language $I \subseteq \Sigma^*$ is called a *two-sided ideal* (or simply an *ideal*) if $I \neq \emptyset$ and $I = \Sigma^* I \Sigma^*$.
- For every synchronizing automaton $\mathscr{A}$ it holds

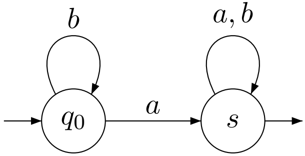
$$\Sigma^* \operatorname{Syn}(\mathscr{A})\Sigma^* = \operatorname{Syn}(\mathscr{A}).$$

Ideals and Synchronizing Automata

- A language $I \subseteq \Sigma^*$ is called a *two-sided ideal* (or simply an *ideal*) if $I \neq \emptyset$ and $I = \Sigma^* I \Sigma^*$.
- For every synchronizing automaton $\mathscr{A}$ it holds

$$\Sigma^* \operatorname{Syn}(\mathscr{A}) \Sigma^* = \operatorname{Syn}(\mathscr{A}).$$

- Every regular ideal language I is a language of reset words for some synchronizing automaton $\mathscr{A}$ (e.g. for its minimal recognizing automaton)



$$L[\mathscr{A}] = I = \Sigma^* a \Sigma^* = \operatorname{Syn}(\mathscr{A})$$

Reset Complexity

Ideal I

Classical

New

$$\mathscr{A}_I = \langle Q_1, \Sigma, \delta_1, q_0, F \rangle$$

$$w \in I \Leftrightarrow q_0.w \in F$$

$$\mathscr{B} = \langle Q_2, \Sigma, \delta_2 \rangle : \mathrm{Syn}(\mathscr{B}) = I$$

$$w \in I \Leftrightarrow |Q_2 \,.\, w| = 1$$

- The *state complexity* $sc(I)$ of a regular language I is the number of sates in the minimal automaton recognizing I.
- The *reset complexity* $rc(I)$ of an ideal language I is the minimal possible number of states in a synchronizing automaton $\mathscr{B}$ such that $\mathrm{Syn}(\mathscr{B}) = I$.
  Every such automaton $\mathscr{B}$ is called *minimal synchronizing automaton* (for brevity, MSA).

Reset Complexity

Ideal I

Classical

New

$$\mathscr{A}_I = \langle Q_1, \Sigma, \delta_1, q_0, F \rangle$$

$$w \in I \Leftrightarrow q_0.w \in F$$

$$\mathscr{B} = \langle Q_2, \Sigma, \delta_2 \rangle : \mathrm{Syn}(\mathscr{B}) = I$$

$$w \in I \Leftrightarrow |Q_2 \, . \, w| = 1$$

- The *state complexity* $sc(I)$ of a regular language I is the number of sates in the minimal automaton recognizing I.
- The *reset complexity* $rc(I)$ of an ideal language I is the minimal possible number of states in a synchronizing automaton $\mathscr{B}$ such that $\mathrm{Syn}(\mathscr{B}) = I$.
  Every such automaton $\mathscr{B}$ is called *minimal synchronizing automaton* (for brevity, MSA).

Reset Complexity

Ideal I

Classical | New

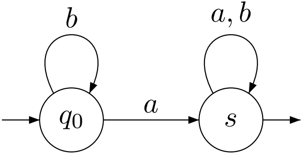
$\mathscr{A}_I = \langle Q_1, \Sigma, \delta_1, q_0, F \rangle$ | $\mathscr{B} = \langle Q_2, \Sigma, \delta_2 \rangle : \mathrm{Syn}(\mathscr{B}) = I$

$w \in I \Leftrightarrow q_0.w \in F$ | $w \in I \Leftrightarrow |Q_2 \,.\, w| = 1$

- The *state complexity* $sc(I)$ of a regular language I is the number of sates in the minimal automaton recognizing I.
- The *reset complexity* $rc(I)$ of an ideal language I is the minimal possible number of states in a synchronizing automaton $\mathscr{B}$ such that $\mathrm{Syn}(\mathscr{B}) = I$.
  Every such automaton $\mathscr{B}$ is called *minimal synchronizing automaton* (for brevity, MSA).

Reset Complexity VS State Complexity

- For every ideal language $rc(I) \leq sc(I)$.



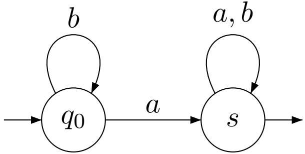
- $L[\mathscr{A}] = I = \Sigma^* a \Sigma^* = \mathrm{Syn}(\mathscr{A})$
- $\mathscr{A}$ is an MSA for I
- $\mathscr{A}$ is the minimal automaton recognizing I
- $rc(I) = sc(I) = 2$

Theorem [M.,2012]

For every $n \geq 3$ there are ideals I_n s.t. $rc(I_n) = n$, and $sc(I_n) = 2^n - n$.

Reset Complexity VS State Complexity

- For every ideal language $rc(I) \leq sc(I)$.



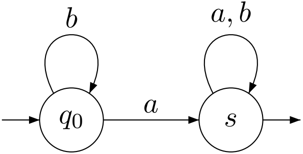
- $L[\mathscr{A}] = I = \Sigma^* a \Sigma^* = \mathrm{Syn}(\mathscr{A})$
- $\mathscr{A}$ is an MSA for I
- $\mathscr{A}$ is the minimal automaton recognizing I
- $rc(I) = sc(I) = 2$

Theorem [M.,2012]

For every $n \geq 3$ there are ideals I_n s.t. $rc(I_n) = n$, and $sc(I_n) = 2^n - n$.

Reset Complexity VS State Complexity

- For every ideal language $rc(I) \le sc(I)$.

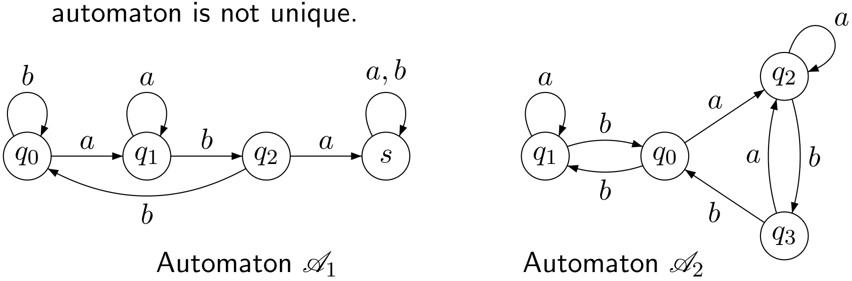


- $L[\mathscr{A}] = I = \Sigma^* a \Sigma^* = \mathrm{Syn}(\mathscr{A})$
- $\mathscr{A}$ is an MSA for I
- $\mathscr{A}$ is the minimal automaton recognizing I
- $rc(I) = sc(I) = 2$

Theorem [M.,2012]

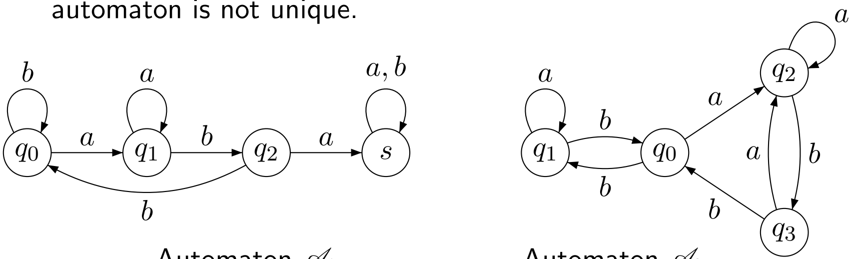
For every $n \ge 3$ there are ideals I_n s.t. $rc(I_n) = n$, and $sc(I_n) = 2^n - n$.

- The representation of an ideal language by an MSA can be exponentially more succinct than using the standard minimal recognizing automaton.
- We do not know how to compute $rc(I)$, and how to build the corresponding automaton. One difficulty is that such automaton is not unique.



- $I = \mathrm{Syn}(\mathscr{A}_1) = \mathrm{Syn}(\mathscr{A}_2) = \Sigma^* aba \Sigma^*$.
- $\mathscr{A}_1$ and $\mathscr{A}_2$ are MSA's for I.

- The representation of an ideal language by an MSA can be exponentially more succinct than using the standard minimal recognizing automaton.
- We do not know how to compute $rc(I)$, and how to build the corresponding automaton. One difficulty is that such automaton is not unique.



Automaton $\mathscr{A}_1$ Automaton $\mathscr{A}_2$

- $I = \mathrm{Syn}(\mathscr{A}_1) = \mathrm{Syn}(\mathscr{A}_2) = \Sigma^* aba \Sigma^*$.
- $\mathscr{A}_1$ and $\mathscr{A}_2$ are MSA's for I.

Main questions

- An ideal I is presented by a DFA $\mathscr{A}$ with $\mathrm{Syn}(\mathscr{A}) = I$. Let $rc(I) \leq k$. It means that there exists some automaton $\mathscr{B}$ with at most k states s.t. $\mathrm{Syn}(\mathscr{B}) = I$.

Question 1

How hard is it to verify the equality $\mathrm{Syn}(\mathscr{A}) = \mathrm{Syn}(\mathscr{B})$?

Question 2

How hard is it to verify the inequality $rc(I) \leq k$?

- DFA's case: the equality $L[\mathscr{A}] = L[\mathscr{B}]$ can be checked easily.
- $\mathrm{Syn}(\mathscr{A})$ and $\mathrm{Syn}(\mathscr{B})$ are regular languages.
- Obstacle: the minimal automaton recognizing $\mathrm{Syn}(\mathscr{A})$ has up to $2^n - n$ states, where n is the size of $\mathscr{A}$.

Main questions

- An ideal I is presented by a DFA $\mathscr{A}$ with $\mathrm{Syn}(\mathscr{A}) = I$. Let $rc(I) \leq k$. It means that there exists some automaton $\mathscr{B}$ with at most k states s.t. $\mathrm{Syn}(\mathscr{B}) = I$.

Question 1

How hard is it to verify the equality $\mathrm{Syn}(\mathscr{A}) = \mathrm{Syn}(\mathscr{B})$?

Question 2

How hard is it to verify the inequality $rc(I) \leq k$?

- DFA's case: the equality $L[\mathscr{A}] = L[\mathscr{B}]$ can be checked easily.
- $\mathrm{Syn}(\mathscr{A})$ and $\mathrm{Syn}(\mathscr{B})$ are regular languages.
- Obstacle: the minimal automaton recognizing $\mathrm{Syn}(\mathscr{A})$ has up to $2^n - n$ states, where n is the size of $\mathscr{A}$.

Main questions

- An ideal I is presented by a DFA $\mathscr{A}$ with $\mathrm{Syn}(\mathscr{A}) = I$. Let $rc(I) \leq k$. It means that there exists some automaton $\mathscr{B}$ with at most k states s.t. $\mathrm{Syn}(\mathscr{B}) = I$.

Question 1

How hard is it to verify the equality $\mathrm{Syn}(\mathscr{A}) = \mathrm{Syn}(\mathscr{B})$?

Question 2

How hard is it to verify the inequality $rc(I) \leq k$?

- DFA's case: the equality $L[\mathscr{A}] = L[\mathscr{B}]$ can be checked easily.
- $\mathrm{Syn}(\mathscr{A})$ and $\mathrm{Syn}(\mathscr{B})$ are regular languages.
- Obstacle: the minimal automaton recognizing $\mathrm{Syn}(\mathscr{A})$ has up to $2^n - n$ states, where n is the size of $\mathscr{A}$.

Main questions

- An ideal I is presented by a DFA $\mathscr{A}$ with $\mathrm{Syn}(\mathscr{A}) = I$. Let $rc(I) \leq k$. It means that there exists some automaton $\mathscr{B}$ with at most k states s.t. $\mathrm{Syn}(\mathscr{B}) = I$.

Question 1

How hard is it to verify the equality $\mathrm{Syn}(\mathscr{A}) = \mathrm{Syn}(\mathscr{B})$?

Question 2

How hard is it to verify the inequality $rc(I) \leq k$?

- DFA's case: the equality $L[\mathscr{A}] = L[\mathscr{B}]$ can be checked easily.
- $\mathrm{Syn}(\mathscr{A})$ and $\mathrm{Syn}(\mathscr{B})$ are regular languages.
- Obstacle: the minimal automaton recognizing $\mathrm{Syn}(\mathscr{A})$ has up to $2^n - n$ states, where n is the size of $\mathscr{A}$.

Main questions

- An ideal I is presented by a DFA $\mathscr{A}$ with $\mathrm{Syn}(\mathscr{A}) = I$. Let $rc(I) \leq k$. It means that there exists some automaton $\mathscr{B}$ with at most k states s.t. $\mathrm{Syn}(\mathscr{B}) = I$.

Question 1

How hard is it to verify the equality $\mathrm{Syn}(\mathscr{A}) = \mathrm{Syn}(\mathscr{B})$?

Question 2

How hard is it to verify the inequality $rc(I) \leq k$?

- DFA's case: the equality $L[\mathscr{A}] = L[\mathscr{B}]$ can be checked easily.
- $\mathrm{Syn}(\mathscr{A})$ and $\mathrm{Syn}(\mathscr{B})$ are regular languages.
- Obstacle: the minimal automaton recognizing $\mathrm{Syn}(\mathscr{A})$ has up to $2^n - n$ states, where n is the size of $\mathscr{A}$.

Main questions

- An ideal I is presented by a DFA $\mathscr{A}$ with $\mathrm{Syn}(\mathscr{A}) = I$. Let $rc(I) \leq k$. It means that there exists some automaton $\mathscr{B}$ with at most k states s.t. $\mathrm{Syn}(\mathscr{B}) = I$.

Question 1

How hard is it to verify the equality $\mathrm{Syn}(\mathscr{A}) = \mathrm{Syn}(\mathscr{B})$?

Question 2

How hard is it to verify the inequality $rc(I) \leq k$?

- DFA's case: the equality $L[\mathscr{A}] = L[\mathscr{B}]$ can be checked easily.
- $\mathrm{Syn}(\mathscr{A})$ and $\mathrm{Syn}(\mathscr{B})$ are regular languages.
- Obstacle: the minimal automaton recognizing $\mathrm{Syn}(\mathscr{A})$ has up to $2^n - n$ states, where n is the size of $\mathscr{A}$.

Problems

SYN-EQUALITY

–*Input:* synchronizing automata $\mathscr{A}$ and $\mathscr{B}$.

–*Question:* is $\mathrm{Syn}(\mathscr{A}) = \mathrm{Syn}(\mathscr{B})$?

RESET-COMPLEXITY ($\leq$)

–*Input:* a synchronizing automaton $\mathscr{A}$, $k \in \mathbb{N}$.

–*Question:* is $rc(I) \leq k$, where $I = \mathrm{Syn}(\mathscr{A})$?

SYN-EQUALITY

–*Input:* synchronizing automata $\mathscr{A}$ and $\mathscr{B}$.

–*Question:* is $\mathrm{Syn}(\mathscr{A}) = \mathrm{Syn}(\mathscr{B})$?

RESET-COMPLEXITY ($\leq$)

–*Input:* a synchronizing automaton $\mathscr{A}$, $k \in \mathbb{N}$.

–*Question:* is $rc(I) \leq k$, where $I = \mathrm{Syn}(\mathscr{A})$?

Belonging to PSPACE

SYN-INCLUSION

–*Input:* synchronizing automata $\mathscr{A}$ and $\mathscr{B}$.

–*Question:* is $\mathrm{Syn}(\mathscr{A}) \subseteq \mathrm{Syn}(\mathscr{B})$?

Theorem 1.

SYN-INCLUSION is in **PSPACE**.

Corollary 1.

SYN-EQUALITY, RESET-COMPLEXITY ($\leq$) are in **PSPACE**.

SYN-INCLUSION

–*Input:* synchronizing automata $\mathscr{A}$ and $\mathscr{B}$.

–*Question:* is $\mathrm{Syn}(\mathscr{A}) \subseteq \mathrm{Syn}(\mathscr{B})$?

Theorem 1.

SYN-INCLUSION is in **PSPACE**.

Corollary 1.

SYN-EQUALITY, RESET-COMPLEXITY ($\leq$) are in **PSPACE**.

Belonging to **PSPACE**

SYN-INCLUSION

–*Input:* synchronizing automata $\mathscr{A}$ and $\mathscr{B}$.

–*Question:* is $\mathrm{Syn}(\mathscr{A}) \subseteq \mathrm{Syn}(\mathscr{B})$?

Theorem 1.

SYN-INCLUSION is in **PSPACE**.

Corollary 1.

SYN-EQUALITY, RESET-COMPLEXITY ($\leq$) are in **PSPACE**.

FINITE AUTOMATA INTERSECTION

–*Input:* given n DFAs $M_i = \langle Q_i, \Sigma, \delta_i, q_i, F_i \rangle$, for $i = 1, \ldots, n$.

–*Question:* is $\bigcap_i L[M_i] \neq \emptyset$?

- It is assumed that $|\Sigma| = 2$.
- We construct the DFA $\mathscr{A} = \langle Q, \Delta, \varphi, \rangle$ with $Q = \bigcup_{i=1}^{n} Q_i \cup \{s, h\}$ and $\Delta = \Sigma \cup \{x, y, z\}$.
- $I = L_1 \cup L_2$; $L_1 = (\Sigma \cup \{x\})^* y \Delta^*$; $L_2 = (\Sigma \cup \{x\})^* z \Delta^+$.

Lemma 1.

$\bigcap_{i=1}^{n} L[M_i] = \emptyset$ iff $\mathrm{Syn}(\mathscr{A}) = I$.

FINITE AUTOMATA INTERSECTION

–*Input:* given n DFAs $M_i = \langle Q_i, \Sigma, \delta_i, q_i, F_i \rangle$, for $i = 1, \ldots, n$.

–*Question:* is $\bigcap_i L[M_i] \neq \emptyset$?

- It is assumed that $|\Sigma| = 2$.
- We construct the DFA $\mathscr{A} = \langle Q, \Delta, \varphi, \rangle$ with $Q = \bigcup_{i=1}^{n} Q_i \cup \{s, h\}$ and $\Delta = \Sigma \cup \{x, y, z\}$.
- $I = L_1 \cup L_2$; $L_1 = (\Sigma \cup \{x\})^* y \Delta^*$; $L_2 = (\Sigma \cup \{x\})^* z \Delta^+$.

Lemma 1.

$\bigcap_{i=1}^{n} L[M_i] = \emptyset$ iff $\mathrm{Syn}(\mathscr{A}) = I$.

FINITE AUTOMATA INTERSECTION

–*Input:* given n DFAs $M_i = \langle Q_i, \Sigma, \delta_i, q_i, F_i \rangle$, for $i = 1, \ldots, n$.
–*Question:* is $\bigcap_i L[M_i] \neq \emptyset$?

- It is assumed that $|\Sigma| = 2$.
- We construct the DFA $\mathscr{A} = \langle Q, \Delta, \varphi, \rangle$ with $Q = \bigcup_{i=1}^{n} Q_i \cup \{s, h\}$ and $\Delta = \Sigma \cup \{x, y, z\}$.
- $I = L_1 \cup L_2$; $L_1 = (\Sigma \cup \{x\})^* y \Delta^*$; $L_2 = (\Sigma \cup \{x\})^* z \Delta^+$.

Lemma 1.

$\bigcap_{i=1}^{n} L[M_i] = \emptyset$ iff $\mathrm{Syn}(\mathscr{A}) = I$.

PSPACE-completeness: automaton $\mathscr{A}$

FINITE AUTOMATA INTERSECTION

–*Input:* given n DFAs $M_i = \langle Q_i, \Sigma, \delta_i, q_i, F_i \rangle$, for $i = 1, \ldots, n$.

–*Question:* is $\bigcap_i L[M_i] \neq \emptyset$?

- It is assumed that $|\Sigma| = 2$.
- We construct the DFA $\mathscr{A} = \langle Q, \Delta, \varphi, \rangle$ with $Q = \bigcup_{i=1}^{n} Q_i \cup \{s, h\}$ and $\Delta = \Sigma \cup \{x, y, z\}$.
- $I = L_1 \cup L_2$; $L_1 = (\Sigma \cup \{x\})^* y \Delta^*$; $L_2 = (\Sigma \cup \{x\})^* z \Delta^+$.

Lemma 1.

$\bigcap_{i=1}^{n} L[M_i] = \emptyset$ iff $\mathrm{Syn}(\mathscr{A}) = I$.

FINITE AUTOMATA INTERSECTION

–*Input:* given n DFAs $M_i = \langle Q_i, \Sigma, \delta_i, q_i, F_i \rangle$, for $i = 1, \ldots, n$.
–*Question:* is $\bigcap_i L[M_i] \neq \emptyset$?

- It is assumed that $|\Sigma| = 2$.
- We construct the DFA $\mathscr{A} = \langle Q, \Delta, \varphi, \rangle$ with $Q = \bigcup_{i=1}^{n} Q_i \cup \{s, h\}$ and $\Delta = \Sigma \cup \{x, y, z\}$.
- $I = L_1 \cup L_2$; $L_1 = (\Sigma \cup \{x\})^* y \Delta^*$; $L_2 = (\Sigma \cup \{x\})^* z \Delta^+$.

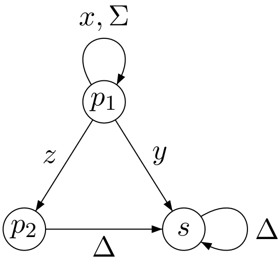
Lemma 1.

$\bigcap_{i=1}^{n} L[M_i] = \emptyset$ iff $\mathrm{Syn}(\mathscr{A}) = I$.

PSPACE-completeness: automaton $\mathscr{B}$

Lemma 2.

$\mathrm{Syn}(\mathscr{B}) = I$.



$$I = L_1 \cup L_2;\ L_1 = (\Sigma \cup \{x\})^* y \Delta^*;\ L_2 = (\Sigma \cup \{x\})^* z \Delta^+.$$

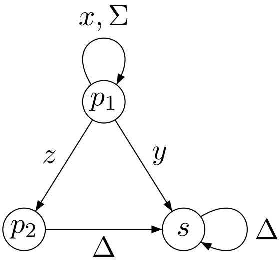
Lemma 3.

$\bigcap_{i=1}^{n} L[M_i] = \emptyset$ iff $\mathrm{Syn}(\mathscr{A}) = \mathrm{Syn}(\mathscr{B})$.

PSPACE-completeness: automaton $\mathscr{B}$

Lemma 2.

$\mathrm{Syn}(\mathscr{B}) = I$.



$I = L_1 \cup L_2$; $L_1 = (\Sigma \cup \{x\})^* y \Delta^*$; $L_2 = (\Sigma \cup \{x\})^* z \Delta^+$.

Lemma 3.

$\bigcap_{i=1}^{n} L[M_i] = \emptyset$ iff $\mathrm{Syn}(\mathscr{A}) = \mathrm{Syn}(\mathscr{B})$.

PSPACE-completeness

Theorem 2.

SYN-EQUALITY is **PSPACE**-complete.

- Polynomial reduction from the negation of FINITE AUTOMATA INTERSECTION to SYN-EQUALITY.
- $\mathscr{B}$ is an MSA for I.
- $\mathscr{B}$ is a particular 3-state synchronizing automaton with a sink state.

Theorem 2.

SYN-EQUALITY is **PSPACE**-complete.

- Polynomial reduction from the negation of FINITE AUTOMATA INTERSECTION to SYN-EQUALITY.
- $\mathscr{B}$ is an MSA for I.
- $\mathscr{B}$ is a particular 3-state synchronizing automaton with a sink state.

Theorem 2.

SYN-EQUALITY is **PSPACE**-complete.

- Polynomial reduction from the negation of FINITE AUTOMATA INTERSECTION to SYN-EQUALITY.
- $\mathscr{B}$ is an MSA for I.
- $\mathscr{B}$ is a particular 3-state synchronizing automaton with a sink state.

Checking the inequality $rc(I) \leq k$

- Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ and $I = \mathrm{Syn}(\mathscr{A})$.
- The inequality $rc(I) \leq 2$ can be checked in polynomial of the size of $\mathscr{A}$ time.
- RESET-COMPLEXITY $(\leq)$ is in **PSPACE**.
- We have constructed for each instance of FINITE AUTOMATA INTERSECTION the corresponding automaton $\mathscr{A}$. Let $I = \mathrm{Syn}(\mathscr{A})$.
- If $\bigcap_{i=1}^{n} L[M_i] = \emptyset$, then $I = J$, where J is the language of reset words of a 3-state automaton $\mathscr{B}$. In this case $rc(I) \leq 3$.
- If $\bigcap_{i=1}^{n} L[M_i] \neq \emptyset$, then I does not serve as the language of reset words for some automaton $\mathscr{B}$ with at most three states. In this case $rc(I) > 3$.

Theorem 3.

RESET-COMPLEXITY $(\leq)$ is **PSPACE**-complete.

Checking the inequality $rc(I) \leq k$

- Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ and $I = \mathrm{Syn}(\mathscr{A})$.
- The inequality $rc(I) \leq 2$ can be checked in polynomial of the size of $\mathscr{A}$ time.
- RESET-COMPLEXITY ($\leq$) is in **PSPACE**.
- We have constructed for each instance of FINITE AUTOMATA INTERSECTION the corresponding automaton $\mathscr{A}$. Let $I = \mathrm{Syn}(\mathscr{A})$.
- If $\bigcap_{i=1}^{n} L[M_i] = \emptyset$, then $I = J$, where J is the language of reset words of a 3-state automaton $\mathscr{B}$. In this case $rc(I) \leq 3$.
- If $\bigcap_{i=1}^{n} L[M_i] \neq \emptyset$, then I does not serve as the language of reset words for some automaton $\mathscr{B}$ with at most three states. In this case $rc(I) > 3$.

Theorem 3.

RESET-COMPLEXITY ($\leq$) is **PSPACE**-complete.

Checking the inequality $rc(I) \leq k$

- Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ and $I = \mathrm{Syn}(\mathscr{A})$.
- The inequality $rc(I) \leq 2$ can be checked in polynomial of the size of $\mathscr{A}$ time.
- RESET-COMPLEXITY ($\leq$) is in **PSPACE**.
- We have constructed for each instance of FINITE AUTOMATA INTERSECTION the corresponding automaton $\mathscr{A}$. Let $I = \mathrm{Syn}(\mathscr{A})$.
- If $\bigcap_{i=1}^{n} L[M_i] = \emptyset$, then $I = J$, where J is the language of reset words of a 3-state automaton $\mathscr{B}$. In this case $rc(I) \leq 3$.
- If $\bigcap_{i=1}^{n} L[M_i] \neq \emptyset$, then I does not serve as the language of reset words for some automaton $\mathscr{B}$ with at most three states. In this case $rc(I) > 3$.

Theorem 3.

RESET-COMPLEXITY ($\leq$) is **PSPACE**-complete.

Checking the inequality $rc(I) \leq k$

- Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ and $I = \mathrm{Syn}(\mathscr{A})$.
- The inequality $rc(I) \leq 2$ can be checked in polynomial of the size of $\mathscr{A}$ time.
- RESET-COMPLEXITY $(\leq)$ is in **PSPACE**.
- We have constructed for each instance of FINITE AUTOMATA INTERSECTION the corresponding automaton $\mathscr{A}$. Let $I = \mathrm{Syn}(\mathscr{A})$.
- If $\bigcap_{i=1}^{n} L[M_i] = \emptyset$, then $I = J$, where J is the language of reset words of a 3-state automaton $\mathscr{B}$. In this case $rc(I) \leq 3$.
- If $\bigcap_{i=1}^{n} L[M_i] \neq \emptyset$, then I does not serve as the language of reset words for some automaton $\mathscr{B}$ with at most three states. In this case $rc(I) > 3$.

Theorem 3.

RESET-COMPLEXITY $(\leq)$ is **PSPACE**-complete.

Checking the inequality $rc(I) \leq k$

- Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ and $I = \mathrm{Syn}(\mathscr{A})$.
- The inequality $rc(I) \leq 2$ can be checked in polynomial of the size of $\mathscr{A}$ time.
- RESET-COMPLEXITY ($\leq$) is in **PSPACE**.
- We have constructed for each instance of FINITE AUTOMATA INTERSECTION the corresponding automaton $\mathscr{A}$. Let $I = \mathrm{Syn}(\mathscr{A})$.
- If $\bigcap_{i=1}^{n} L[M_i] = \emptyset$, then $I = J$, where J is the language of reset words of a 3-state automaton $\mathscr{B}$. In this case $rc(I) \leq 3$.
- If $\bigcap_{i=1}^{n} L[M_i] \neq \emptyset$, then I does not serve as the language of reset words for some automaton $\mathscr{B}$ with at most three states. In this case $rc(I) > 3$.

Theorem 3.

RESET-COMPLEXITY ($\leq$) is **PSPACE**-complete.

Checking the inequality $rc(I) \leq k$

- Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ and $I = \mathrm{Syn}(\mathscr{A})$.
- The inequality $rc(I) \leq 2$ can be checked in polynomial of the size of $\mathscr{A}$ time.
- RESET-COMPLEXITY ($\leq$) is in **PSPACE**.
- We have constructed for each instance of FINITE AUTOMATA INTERSECTION the corresponding automaton $\mathscr{A}$. Let $I = \mathrm{Syn}(\mathscr{A})$.
- If $\bigcap_{i=1}^{n} L[M_i] = \emptyset$, then $I = J$, where J is the language of reset words of a 3-state automaton $\mathscr{B}$. In this case $rc(I) \leq 3$.
- If $\bigcap_{i=1}^{n} L[M_i] \neq \emptyset$, then I does not serve as the language of reset words for some automaton $\mathscr{B}$ with at most three states. In this case $rc(I) > 3$.

Theorem 3.

RESET-COMPLEXITY ($\leq$) is **PSPACE**-complete.

Future work

- Computational complexity of RESET-COMPLEXITY ($\leq$) for a non-unary alphabet of size less than five.
- Studying the reset complexity w.r.t. boolean operations: intersection, union, concatenation.
- Representation of ideal languages by non-deterministic finite automata.

Future work

- Computational complexity of RESET-COMPLEXITY ($\leq$) for a non-unary alphabet of size less than five.
- Studying the reset complexity w.r.t. boolean operations: intersection, union, concatenation.
- Representation of ideal languages by non-deterministic finite automata.

Future work

- Computational complexity of RESET-COMPLEXITY ($\leq$) for a non-unary alphabet of size less than five.
- Studying the reset complexity w.r.t. boolean operations: intersection, union, concatenation.
- Representation of ideal languages by non-deterministic finite automata.