Minicomplexity

Survey

Aliya Khadieva. Supervisor: Abuzer Yakaryılmaz

University of Latvia

2018/06/13

Summary

LATVIJAS UNIVERSITATE ANNO 1919

1-way deterministic finite automaton (1DFA)

Definition

A **1-way DFA** is a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$

- Q is a finite set of states
- Σ is an input alphabet
- $\delta : Q \times \Sigma \to Q$ is transition function
- $q_0 \in Q$ is a starting state
- $F \subseteq Q$ is a set of accepting states

The set of strings that M accepts is the **language** recognized by M and this language is denoted by $L(M)$.

1-way deterministic finite automaton (1DFA)

Definition

A **1-way DFA** is a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$

- Q is a finite set of states
- Σ is an input alphabet
- $\delta : Q \times \Sigma \to Q$ is transition function
- $q_0 \in Q$ is a starting state
- $F \subseteq Q$ is a set of accepting states

The set of strings that M accepts is the **language** recognized by M and this language is denoted by $L(M)$.

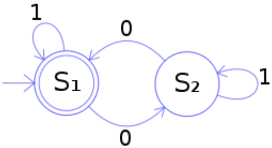


FIGURE – a DFA M, with a binary alphabet, which requires that the input contains an even number of 0s

LATVIJAS UNIVERSITATE ANNO 1919

1-way deterministic finite automaton (1DFA)

Definition

A **1-way DFA** is a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$

- Q is a finite set of states
- Σ is an input alphabet
- $\delta : Q \times \Sigma \to Q$ is transition function
- $q_0 \in Q$ is a starting state
- $F \subseteq Q$ is a set of accepting states

The set of strings that M accepts is the **language** recognized by M and this language is denoted by $L(M)$.

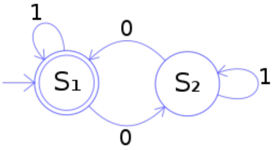


FIGURE – a DFA M, with a binary alphabet, which requires that the input contains an even number of 0s

Industrial programming applications :

- state machine pattern
- streaming algorithms
- 1-way and 2-way lists processing

TATE

1-way deterministic finite automaton

MEMBERSHIP **problem**.
Let $[h] = \{0, \ldots, h-1\}$, $[|h|]$ is a set of all subsets of $[h]$.
Let $h \geq 1$, consider the (promise) problem over $\Sigma = [h] \cup [|h|]$:
Given a number $i \in [h]$ and a set $\alpha \subseteq [h]$, check that $i \in \alpha$

every instance is promised to be of form : $\vdash i \quad \alpha \dashv$

1-way deterministic finite automaton

MEMBERSHIP **problem**.
Let $[h] = \{0, \ldots, h-1\}$, $[|h|]$ is a set of all subsets of $[h]$.
Let $h \geq 1$, consider the (promise) problem over $\Sigma = [h] \cup [|h|]$:
Given a number $i \in [h]$ and a set $\alpha \subseteq [h]$, check that $i \in \alpha$

every instance is promised to be of form : $\vdash i \quad \alpha \dashv$

- From state 0 on $\vdash$, move to 0 on the 1st cell.
- Reading i, move to state i on the 2nd cell.
- Reading α in state i, check whether $i \in \alpha$.
- If not, then hang. Otherwise, move to state 0 on $\dashv$. Then fall off $\dashv$, again in state 0.

LATVIJAS UNIVERSITATE ANNO 1919

1-way deterministic finite automaton

MEMBERSHIP **problem**.
Let $[h] = \{0, \ldots, h-1\}$, $[|h|]$ is a set of all subsets of $[h]$.
Let $h \geq 1$, consider the (promise) problem over $\Sigma = [h] \cup [|h|]$:
Given a number $i \in [h]$ and a set $\alpha \subseteq [h]$, check that $i \in \alpha$

> every instance is promised to be of form :
> $\vdash i \quad \alpha \dashv$

- From state 0 on $\vdash$, move to 0 on the 1st cell.
- Reading i, move to state i on the 2nd cell.
- Reading α in state i, check whether $i \in \alpha$.
- If not, then hang. Otherwise, move to state 0 on $\dashv$. Then fall off $\dashv$, again in state 0.

1DFA $M = ([h], \Sigma, ., 0, 0)$ **uses** h **states!**

LATVIJAS UNIVERSITATE ANNO 1919

1-way deterministic finite automaton

MEMBERSHIP^R **problem**.
The reverse of *MEMBERSHIP* problem.
Given a number set $\alpha \subseteq [h]$, $i \in [h]$, check that $i \in \alpha$

every instance is promised to be of form : $\vdash \alpha \quad i \dashv$

LATVIJAS UNIVERSITĀTE ANNO 1919

1-way deterministic finite automaton

MEMBERSHIP^R **problem**.
The reverse of *MEMBERSHIP* problem.
Given a number set $\alpha \subseteq [h]$, $i \in [h]$, check that $i \in \alpha$

every instance is promised to be of form :
$\vdash \alpha \quad i \dashv$

- From state $\emptyset$ on $\vdash$, move to $\emptyset$ on the 1st cell.
- Reading α, move to state α on the 2nd cell.
- Reading i in state α, check whether $i \in \alpha$.
- If not, then hang. Otherwise, move to state $\emptyset$ on $\dashv$. Then fall off $\dashv$, again in state $\emptyset$.

LATVIJAS UNIVERSITATE ANNO 1919

1-way deterministic finite automaton

MEMBERSHIP^R **problem**.
The reverse of *MEMBERSHIP* problem.
Given a number set $\alpha \subseteq [h]$, $i \in [h]$, check that $i \in \alpha$

every instance is promised to be of form : $\vdash \alpha \quad i \dashv$

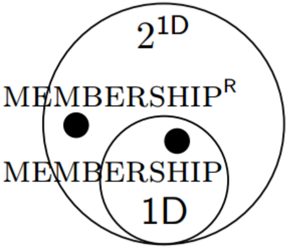
- From state $\emptyset$ on $\vdash$, move to $\emptyset$ on the 1st cell.
- Reading α, move to state α on the 2nd cell.
- Reading i in state α, check whether $i \in \alpha$.
- If not, then hang. Otherwise, move to state $\emptyset$ on $\dashv$. Then fall off $\dashv$, again in state $\emptyset$.

1DFA $M = ([|h|], \Sigma, ., \emptyset, \emptyset)$ **uses** 2^h **states !**

LATVIJAS UNIVERSITATE ANNO 1919

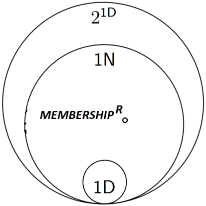
Classes $1D$ and 2^{1D}

- $1D := \{(\mathcal{L}_h)_{h\geq 1}$ |for some polynomial p and 1DFA family $(M_h)_{h\geq 1}$, every M_h solves $\mathcal{L}$ with $\leq p(h)$ states$\}$
- $2^{1D} := \{(\mathcal{L}_h)_{h\geq 1}$ |for some polynomial p and 1DFA family $(M_h)_{h\geq 1}$, every M_h solves $\mathcal{L}$ with $\leq 2^{p(h)}$ states$\}$



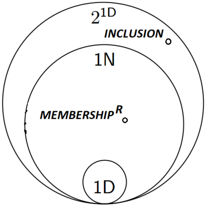
Class 1N

$1D \subseteq 1N \subseteq 2^{1D}$



Class 1N

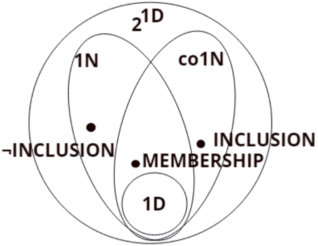
- Consider the *INCLUSION* problem defined over $[|h|]$.
- Given two sets $\alpha, \beta \subseteq [h]$, check that $\alpha \subseteq \beta$
- Instance is promised to be of the form : $\vdash \alpha \quad \beta \dashv$
- $INCLUSION \in 2^{1D}/1N$



Class co1N

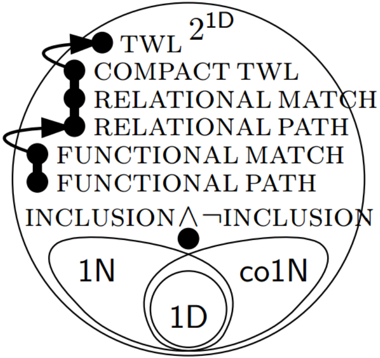
There are some problems that do not admit small 1NFAs, their complements do.
E.g., $\neg INCLUSION_h$ is solved by an $(h+1)$-state 1NFA which 'guesses' an $i \in \alpha \backslash \beta$

$co1N := \{(\mathcal{L}_h)_{h\geq 1} \mid$ for some polynomial p and 1NFA family $(M_h)_{h\geq 1}$,
every M_h solves $\neg\mathcal{L}$ with $\leq p(h)$ states$\}$



Harder problems $\in 2^{1D}/(1N \cup co1N)$

For a problem in $2^{1D}/(1N \cup co1N)$, we may consider the conjunctive concatenation of any two problems from the two sides of the symmetric difference of 1N and co1N, e.g., $INCLUSION \wedge \neg INCLUSION$

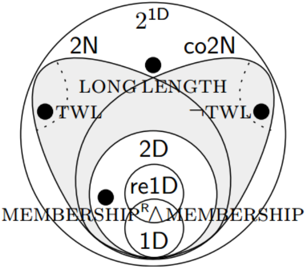


2-way automata

$1D \subseteq 2D \subseteq 2N \subseteq 2^{1D}$
$1D, re1D \subsetneq 2D \subseteq 2N, co2N \subsetneq 2^{1D}$

The Sakoda-Sipser conjecture

$2D \neq 2N$

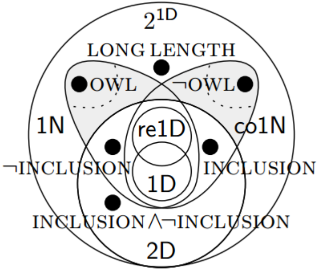


2-way automata

$1D \subseteq 2D \subseteq 2N \subseteq 2^{1D}$
$1D, re1D \subsetneq 2D \subseteq 2N, co2N \subsetneq 2^{1D}$

A stronger conjecture

$1N \not\subseteq 2D$



Hardness propagation technique

For proving separation between the complexity classes.

Classes separating technique

To separate two complexity classes C_1 and C_2 it is suffice to provide **witness**. That is a family of languages $\mathcal{L} \in C_2/C_1$.
Finding $\mathcal{L} \in C_2$ is easy.
$\mathcal{L} \notin C_1$ is more difficult.

Hardness propagation technique

For proving separation between the complexity classes.

Classes separating technique

To separate two complexity classes C_1 and C_2 it is suffice to provide **witness**. That is a family of languages $\mathcal{L} \in C_2/C_1$.
Finding $\mathcal{L} \in C_2$ is easy.
$\mathcal{L} \notin C_1$ is more difficult.

Let C_0 be very restricted.
$\mathcal{L} \notin C_0$.
If we find a language family operator $\mathcal{O}$ such that

- For each $\mathcal{L} \notin C_0 \rightarrow \mathcal{O}(\mathcal{L}) \notin C_1$,
- C_2 is closed under operator $\mathcal{O}$,

then we obtain a witness $\mathcal{O}(\mathcal{L}) \in C_2/C_1$.
The operator $\mathcal{O}$ **propagates hardness** from $\mathcal{L}$ vs. C_0 to $\mathcal{O}(\mathcal{L})$ vs. C_1.

LATVIJAS UNIVERSITATE ANNO 1919

Our future plans

- To consider and prepare a survey on complexity classes of languages recognizable by $1PFAs$ ($2PFAs$), $1QFAs$.
- To find the correlations between corresponding classes using classes separating techniques.
- To come up with different hardness propagation techniques.
- To consider the alternations and polynomial-size hierarchies of complexity.

Quantum finite automata

Definition

1-way QFA is a tuple $M = (Q, \Sigma, \delta, q_0, Q_{acc}, Q_{rej})$

- Q is a finite set of states
- Σ is an input alphabet
- δ is transition function
- $q_0 \in Q$ is a starting state
- Q_{acc}, Q_{rej} are sets of accepting and rejecting states
- & and $ are the left and the right endmarkers, do not belong to Σ

- Working alphabet is $\Gamma = \Sigma \cup \{\&, \$\}$
- A superposition of M is any element of $l_2(Q)$ (the space of mapping from Q to $\mathbb{C}$)
- For $q \in Q$, $|q\rangle$ is unit vector with value 1 at q and 0 elsewhere
- All elements ψ of $l_2(Q)$ - combinations of vectors $|q\rangle$

Quantum finite automata

Transition function δ maps $Q \times \Gamma \times Q$ to $\mathbb{C}$

For $a \in \Gamma$ V_a is a liner transformation on $l_2(Q)$ defined by
$$V_a(|q_1\rangle) = \sum_{q_2 \in Q} \delta(q_1, a, q_2)|q_2\rangle$$

LATVIJAS UNIVERSITATE ANNO 1919

Quantum finite automata

Transition function δ maps $Q \times \Gamma \times Q$ to $\mathbb{C}$

For $a \in \Gamma$ V_a is a liner transformation on $l_2(Q)$ defined by
$$V_a(|q_1\rangle) = \sum_{q_2 \in Q} \delta(q_1, a, q_2)|q_2\rangle$$

After reading the right endmarker \$ ψ is obsrved with respect to $E_{acc} \oplus E_{rej}$
$E_{acc} = span\{|q\rangle : q \in Q_{acc}\}$
$E_{rej} = span\{|q\rangle : q \in Q_{rej}\}$
Observation gives $x \in E_i$ with probability equal to square of the projection of ψ to E_i.
If $\psi \in E_{acc}$, the input is accepted, otherwise is rejected

Previous results on QFAs

- Let p be a prime
- $L_p = \{a^j \mid j \text{ is divisible by } p\}$

Previous results on QFAs

- Let p be a prime
- $L_p = \{a^j \mid j \text{ is divisible by } p\}$

Any 1-way *DFA* recognizing L_p has at least p states.

There is a much more efficient *QFA*

Previous results on QFAs

- Let p be a prime
- $L_p = \{a^j \mid j \text{ is divisible by } p\}$

Any 1-way *DFA* recognizing L_p has at least p states.

There is a much more efficient *QFA*

Ambainis, Freivalds. 1998

L_p can be recognized by a *QFA* with $O(\log p)$ states for prime p. .
Big-O constant depends on required probability of correct answer.

Ambainis, Nahimovs. 2009

For any $\epsilon > 0$, there is a *QFA* with $4\frac{\log 2p}{\epsilon}$ states recognizing L_p with probability at least $1 - \epsilon$ for prime p.

Ablayev, Vasiliev. 2010

For any $\epsilon > 0$, there is a *QFA* with $2\frac{\ln 2p}{\epsilon}$ states recognizing L_p with probability at least $1 - \epsilon$ for any integer p.

LATVIJAS UNIVERSITATE ANNO 1919

Thank you for attention !