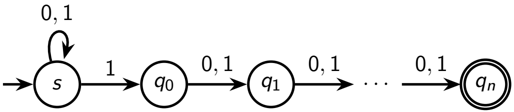
Fine-grained Complexity of Ambiguity Testing of Automata

Anita Dürr
ETH Zürich

Joint work with Karolina Drabik, Fabian Frei, Filip Mazowiecki and Karol Węgrzycki
Warsaw University, CSAIL MIT, Max Planck Institute for Informatics

Finite Automata

Finite Automaton



$$\mathcal{A} = (Q, \delta, \Sigma, \{s\}, \{q_n\})$$
$$\mathcal{L}(\mathcal{A}) = (0+1)^*1(0+1)^n$$

- A word $w \in \Sigma^*$ is accepted by $\mathcal{A}$ if it labels an accepting run

DFA

✓ efficient for complementation, minimization, acceptance, containment, equivalence ...

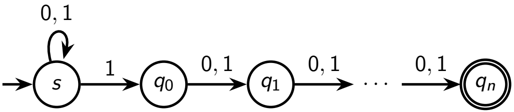
NFA (this talk: no ε-transitions)

✗ containment and equivalence are PSPACE-complete

✓ exponentially more succint

Finite Automata

Finite Automaton



$$\mathcal{A} = (Q, \delta, \Sigma, \{s\}, \{q_n\})$$
$$\mathcal{L}(\mathcal{A}) = (0+1)^*1(0+1)^n$$

- A word $w \in \Sigma^*$ is accepted by $\mathcal{A}$ if it labels an accepting run

DFA

✓ efficient for complementation, minimization, acceptance, containment, equivalence ...

- words label a unique accepting run

NFA (this talk: no ε-transitions)

✗ containment and equivalence are PSPACE-complete

✓ exponentially more succint

- words can label multiple accepting runs

Ambiguity of Finite Automata

Definition

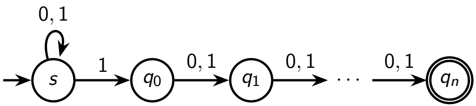
An NFA $\mathcal{A}$ is:

- **unambiguous** if every word has $\leqslant 1$ accepting run. $\rightarrow$ UFA

UFA

✓ UFA is more succint than DFA

✓ containment and equivalence are still polytime in UFA



Ambiguity of Finite Automata

Definition

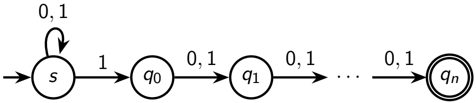
An NFA $\mathcal{A}$ is:

- **unambiguous** if every word has $\leqslant 1$ accepting run. $\rightarrow$ UFA
- **finitely ambiguous** if every word has $\leqslant$ constant accepting runs. $\rightarrow$ FFA
- **polynomially ambiguous** if every word has $\leqslant$ poly(|word|) accepting runs. $\rightarrow$ PFA

UFA

✓ UFA is more succint than DFA

✓ containment and equivalence are still polytime in UFA



$$DFA \subseteq UFA \subseteq FFA \subseteq PFA \subseteq NFA$$

Ambiguity Testing

Problem (Ambiguity Testing)

Given an NFA $\mathcal{A}$, is $\mathcal{A}$ unambiguous / finitely ambiguous / polynomially ambiguous?

$$DFA \subseteq UFA \subseteq FFA \subseteq PFA \subseteq NFA$$

Ambiguity Testing

Problem (Ambiguity Testing)

Given an NFA $\mathcal{A}$, is $\mathcal{A}$ unambiguous / finitely ambiguous / polynomially ambiguous?

$$DFA \subseteq UFA \subseteq FFA \subseteq PFA \subseteq NFA$$

Deciding that $\mathcal{A}$ is	**Running time** [1991, 2001]
unambiguous (UFA)	n^2
finitely ambiguous (FFA)	n^3
polynomially ambiguous (PFA)	n^2

$n = |Q| + |\delta| =$ size of $\mathcal{A}$

Ambiguity Testing

Problem (Ambiguity Testing)

Given an NFA $\mathcal{A}$, is $\mathcal{A}$ unambiguous / finitely ambiguous / polynomially ambiguous?

$$DFA \subseteq UFA \subseteq FFA \subseteq PFA \subseteq NFA$$

Deciding that $\mathcal{A}$ is	**Running time** [1991, 2001]	**Tight under** [our work]
unambiguous (UFA)	n^2	OV or k-cycle
finitely ambiguous (FFA)	n^3	3-OV
polynomially ambiguous (PFA)	n^2	OV or k-cycle

$n = |Q| + |\delta| =$ size of $\mathcal{A}$

Ambiguity Testing

Problem (Ambiguity Testing)

Given an NFA $\mathcal{A}$, is $\mathcal{A}$ unambiguous / finitely ambiguous / polynomially ambiguous?

$$DFA \subseteq UFA \subseteq FFA \subseteq PFA \subseteq NFA$$

Deciding that $\mathcal{A}$ is	**Running time** [1991, 2001]	**Tight under** [our work]
unambiguous (UFA)	n^2	OV or k-cycle
finitely ambiguous (FFA)	n^3	3-OV
polynomially ambiguous (PFA)	n^2	OV or k-cycle

$n = |Q| + |\delta| =$ size of $\mathcal{A}$

An SETH lower bound

An SETH lower bound

Problem (DISJOINTNESS)

Given two DFAs $\mathcal{D}_1$ and $\mathcal{D}_2$, decide whether $\mathcal{L}(\mathcal{D}_1 \cap \mathcal{D}_2) = \varnothing$.

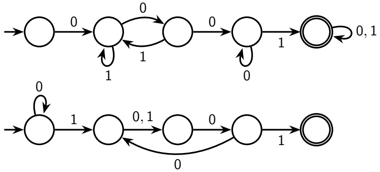
An SETH lower bound

Problem (DISJOINTNESS)

Given two DFAs $\mathcal{D}_1$ and $\mathcal{D}_2$, decide whether $\mathcal{L}(\mathcal{D}_1 \cap \mathcal{D}_2) = \varnothing$.

Disjointness → Unambiguity

- Given $\mathcal{D}_1$ and $\mathcal{D}_2$, let $\mathcal{A} \coloneqq \mathcal{D}_1 \cup \mathcal{D}_2$.
- $\mathcal{A}$ is unambiguous $\Leftrightarrow \mathcal{L}(\mathcal{D}_1 \cap \mathcal{D}_2) = \varnothing$



Improved lower bounds

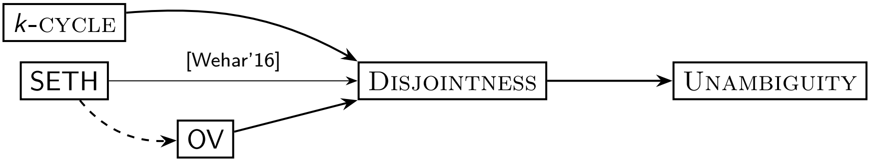
Orthogonal Vector

Given sets $A, B \subseteq \{0,1\}^d$ of size n decide if

$$\exists a \in A, \exists b \in B: \ a \perp b.$$

Hypothesis: $\forall \varepsilon > 0, \exists c > 0$ not in time $\mathcal{O}(n^{2-\varepsilon})$ for $d = c \log n$

Improved lower bounds



k-cycle

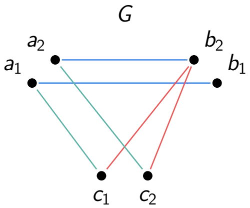
Given a k-partite graph $G = (A_1 \uplus \cdots \uplus A_k, E)$, decide if it contains a cycle of length k.
Hypothesis: $\forall \varepsilon > 0, \exists k \in \mathbb{N}$ not in time $\mathcal{O}(|E|^{2-\varepsilon})$

k-Cycle $\rightarrow$ Disjointness

Construction Given k-partite graph $G = (A_1 \uplus \cdots \uplus A_k, E)$, need to detect k-cycle.

Example
$k = 3$

$n = 2$



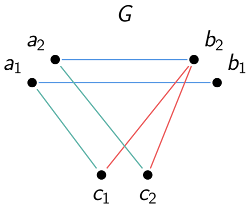
k-Cycle $\to$ Disjointness

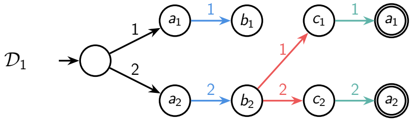
Construction Given k-partite graph $G = (A_1 \uplus \cdots \uplus A_k, E)$, need to detect k-cycle.

- $\mathcal{D}_1$ recognize paths $A_1 \to A_2 \cdots \to A_k \to A_1$,

Example
$k = 3$

$n = 2$





k-Cycle → Disjointness

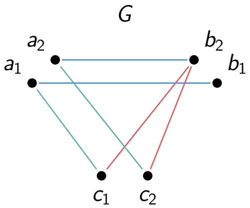
Construction Given k-partite graph $G = (A_1 \uplus \cdots \uplus A_k, E)$, need to detect k-cycle.

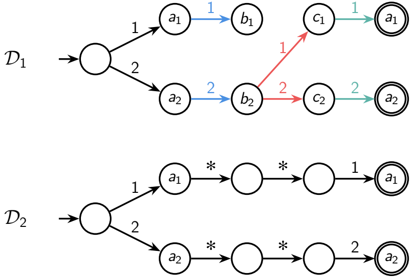
- $\mathcal{D}_1$ recognize paths $A_1 \to A_2 \cdots \to A_k \to A_1$,
- $\mathcal{D}_2$ recognize word of length $k+1$ starting and ending with the same vertex $a \in A_1$.

Example

$k = 3$

$n = 2$





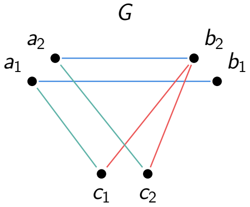
k-Cycle → Disjointness

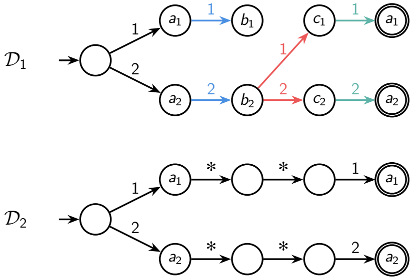
Construction Given k-partite graph $G = (A_1 \uplus \cdots \uplus A_k, E)$, need to detect k-cycle.

- $\mathcal{D}_1$ recognize paths $A_1 \to A_2 \cdots \to A_k \to A_1$,
- $\mathcal{D}_2$ recognize word of length $k+1$ starting and ending with the same vertex $a \in A_1$.

→ Then any word $w \in \mathcal{L}(\mathcal{D}_1) \cap \mathcal{L}(\mathcal{D}_2)$ encodes a k-cycle in G.

Example
$k = 3$
$n = 2$





Size $\mathcal{O}(n + m)$ states and transitions

k-Cycle $\to$ Disjointness

Construction Given k-partite graph $G = (A_1 \uplus \cdots \uplus A_k, E)$, need to detect k-cycle.

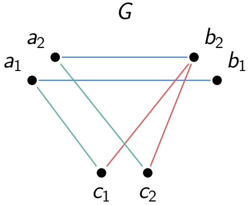
- $\mathcal{D}_1$ recognize paths $A_1 \to A_2 \cdots \to A_k \to A_1$,
- $\mathcal{D}_2$ recognize word of length $k+1$ starting and ending with the same vertex $a \in A_1$.

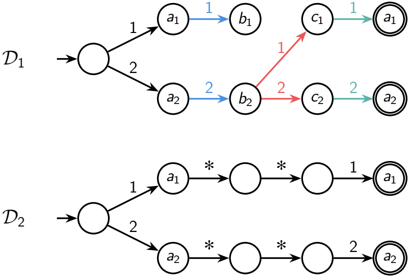
$\to$ Then any word $w \in \mathcal{L}(\mathcal{D}_1) \cap \mathcal{L}(\mathcal{D}_2)$ encodes a k-cycle in G.

Example

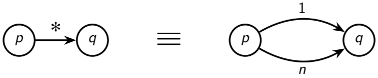
$k = 3$

$n = 2$





Issue



k-Cycle $\rightarrow$ Disjointness

Construction Given k-partite graph $G = (A_1 \uplus \cdots \uplus A_k, E)$, need to detect k-cycle.

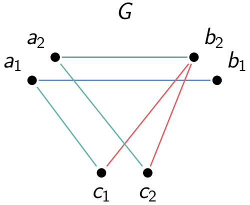
- $\mathcal{D}_1$ recognize paths $A_1 \rightarrow A_2 \cdots \rightarrow A_k \rightarrow A_1$,
- $\mathcal{D}_2$ recognize word of length $k+1$ starting and ending with the same vertex $a \in A_1$.

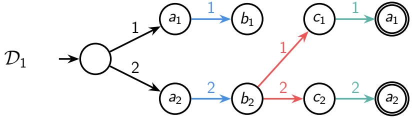
$\rightarrow$ Then any word $w \in \mathcal{L}(\mathcal{D}_1) \cap \mathcal{L}(\mathcal{D}_2)$ encodes a k-cycle in G.

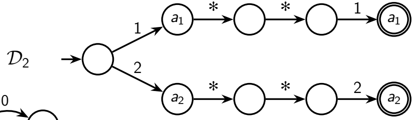
Example

$k = 3$

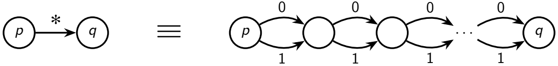
$n = 2$







Fix



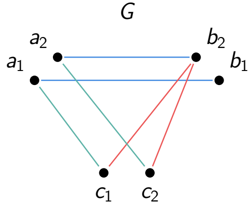
k-Cycle $\rightarrow$ Disjointness

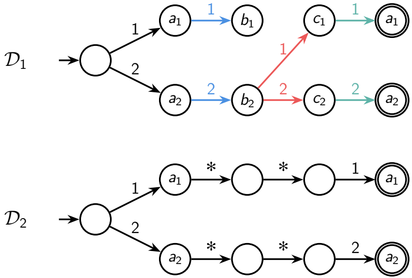
Construction Given k-partite graph $G = (A_1 \uplus \cdots \uplus A_k, E)$, need to detect k-cycle.

- $\mathcal{D}_1$ recognize paths $A_1 \rightarrow A_2 \cdots \rightarrow A_k \rightarrow A_1$,
- $\mathcal{D}_2$ recognize word of length $k+1$ starting and ending with the same vertex $a \in A_1$.

$\rightarrow$ Then any word $w \in \mathcal{L}(\mathcal{D}_1) \cap \mathcal{L}(\mathcal{D}_2)$ encodes a k-cycle in G.

Example
$k = 3$
$n = 2$





Size $\mathcal{O}((n+m)\log n)$ states and transitions

Recap

Problem (Ambiguity Testing)

Given an NFA $\mathcal{A}$, is $\mathcal{A}$ unambiguous / finitely ambiguous / polynomially ambiguous?

Deciding that $\mathcal{A}$ is	**Running time** [1991, 2001]	**Tight under** [our work]
unambiguous (UFA)	n^2	OV or k-cycle
finitely ambiguous (FFA)	n^3	3-OV
polynomially ambiguous (PFA)	n^2	OV or k-cycle

$n = |Q| + |\delta| =$ size of $\mathcal{A}$

Recap

Problem (Ambiguity Testing)

Given an NFA $\mathcal{A}$, is $\mathcal{A}$ unambiguous / finitely ambiguous / polynomially ambiguous?

Deciding that $\mathcal{A}$ is	**Running time** [1991, 2001]	**Tight under** [our work]
unambiguous (UFA)	n^2	OV or k-cycle
finitely ambiguous (FFA)	n^3	3-OV
polynomially ambiguous (PFA)	n^2	OV or k-cycle

$n = |Q| + |\delta| =$ size of $\mathcal{A}$

Thank you!