Overview

1. DFA Intersection Problem
2. Complexity of DFA-IE
3. Unary DFA Intersection Problem
4. Complexity of UDFA-IE
5. Open Problems
Problem Statement

Intersection Non-Emptiness for DFA’s

Given a finite list of DFA’s, is there a string that is accepted by all of the DFA’s?

- We denote this problem by DFA-IE.
- We use \( n \) for the total length of the input’s encoding.
- We use \( k \) for the number of DFA’s in the list.
Intersection Non-Emptiness for DFA’s

Given a finite list of DFA’s, is there a string that is accepted by all of the DFA’s?

- Let DFA’s $D_1, D_2, \ldots, D_k$ be given.
- Construct the Cartesian product DFA $\mathcal{D}$.
- If each DFA has at most $m$ states, then the product has at most $m^k$ states.
- $\mathcal{D}$ accepts a string $x \Leftrightarrow D_i$ accepts $x$ for each $i \in [k]$.
- We just need to check if there exists a string that is accepted by $\mathcal{D}$. 
Hardness and Insights

Intersection Non-Emptiness for DFA’s

Given a finite list of DFA’s, is there a string that is accepted by all of the DFA’s?

- It is a classic PSPACE-complete problem [Kozen 1977].
- There are two ways of viewing this problem:
  - Directed Reachability: Can I get from a start state to a final state in the product graph?
  - Constraint Satisfaction: Is there a string that satisfies all of the DFA’s?
- We use $k$-DFA-IE to denote the problem for fixed $k$-many DFA’s.
Overview

1. DFA Intersection Problem
2. Complexity of DFA-IE
3. Unary DFA Intersection Problem
4. Complexity of UDFA-IE
5. Open Problems
Theorem

Solving $k$-DFA-IE is equivalent to simulating a NTM that uses $k \log(n)$ bits of memory.\(^a\)

\(^a\)Originally shown in Kasai, Iwata 1985.

- Let a $k \log(n)$-space bound NTM $M$ be given.
- Let an input string $x$ of length $n$ be given.
- A computation of $M$ on $x$ is a sequence of configurations.
- Each configuration includes the tape content.
Turing Machine Simulation

Theorem

Solving $k$-DFA-IE is equivalent to simulating a NTM that uses $k \log(n)$ bits of memory.$^a$

$^a$Originally shown in Kasai, Iwata 1985.

- The tape is a sequence of $k \log(n)$ bits.
- We can break up this sequence into $k$ regions.
- Each region will consist of $\log(n)$ bits from the tape.
- We build $k$ DFA’s to collectively verify a “computation”.
- We assign one DFA to each region.
XNL-Completeness

- XNL is a parameterized complexity class.
- It consists of parameterized problems that can be solved in non-deterministic $f(k) \log(n)$ space for some function $f$.
- $k$-DFA-IE is XNL-complete under parameter preserving fpt-reductions.
On the Existence of Faster Algorithms

Question

Can we solve \( k \)-DFA-IE in less than \( n^k \) time?

- It has been conjectured that for every \( k \) and \( \varepsilon > 0 \),
  \( \text{NL}_k \not\subset \text{DTIME}(n^{k-\varepsilon}) \) [Kasai, Iwata 1985]
  - With respect to this conjecture, all XNL-complete problems have conditional lower bounds including \( k \)-DFA-IE.
  - In particular, the existence of faster algorithms for \( k \)-DFA-IE would imply that this conjecture is false.
Question

Can we solve $k$-DFA-IE in less than $n^k$ time?

- If $k$-DFA-IE is solvable in $n^{o(k)}$ time, then:
  - There exist faster algorithms for Integer Factorization and Subset Sum [KLV 2003]
  - There exist faster algorithms for SAT and ETH is false [Fernau, Krebs 2016; Wehar 2016]
  - $NL \subsetneq P$ [Wehar 2014; based on KLV 2003]

- If there exists $k$ and $\varepsilon > 0$ such that $k$-DFA-IE is solvable in $O(n^{k-\varepsilon})$ time, then:
  - There exist faster algorithms for SAT and SETH is false
Overview

1. DFA Intersection Problem
2. Complexity of DFA-IE
3. Unary DFA Intersection Problem
4. Complexity of UDFA-IE
5. Open Problems
A unary DFA is simply a DFA over a unary input alphabet.
Such automata consist of a segment and a cycle.
Each final state from a unary DFA can be viewed as a constraint over natural numbers.
The segment has length two and the cycle has length four.
The DFA accepts:

- the string of length 1
- strings with length congruent to 3 mod 4
- strings with length greater than 0 and divisible by 4
Problem Statement

Intersection Non-Emptiness for Unary DFA’s

Given a finite list of unary DFA’s, is there a string that is accepted by all of the DFA’s?

- We denote this problem by UDFA-IE.
- We use \( n \) for the total length of the input’s encoding.
- We use \( k \) for the number of unary DFA’s in the list.
- UDFA-IE is NP-complete [Stockmeyer, Meyer 1973]
- We show that \( k \)-UDFA-IE is W[1]-complete by providing fpt-reductions to and from \( k \)-clique.
Overview

1. DFA Intersection Problem
2. Complexity of DFA-IE
3. Unary DFA Intersection Problem
4. Complexity of UDFA-IE
5. Open Problems
Reduction

$k$-UDFA-IE is $\text{fpt}$-reducible to $k$-clique.

- Let unary DFA’s $D_1, D_2, \ldots, D_k$ be given.
- Using brute force search, we can check if any unary string of length at most $n$ is accepted.
- If not, then we can ignore final states that appear on the initial segments.
- Next, we build a graph $G$ with $O(n)$ vertices such that $G$ has a $k$-clique if and only if the DFA’s have a non-empty intersection.
Reduction

$k$-UDFA-IE is fpt-reducible to $k$-clique.

- The vertices of $G$ are broken up into $k$ independent sets.
- Within each set, the vertices represent final states from the associated DFA.
- Further, each vertex is associated with a modular equation $x \equiv r \mod c$ where:
  - $x$ is a unary input string interpreted as a natural number
  - $c$ is the cycle length from the associated DFA
  - $r$ is the remainder represented by the associated final state
Reduction

$k$-UDFA-IE is fpt-reducible to $k$-clique.

- Two vertices are adjacent if they are from different sets and their associated modular equations form a solvable system.
- A $k$-clique is associated with:
  - A system of $k$ modular equations where every pair of equations has a solution.
- Applying a variation of the Chinese Remainder Theorem for non-relatively prime moduli:
  - A system of $k$ modular equations has a solution if and only if every pair of equations has a solution.
Reduction

$k$-clique is fpt-reducible to $k$-UDFA-IE.

- Let a graph $G$ with $n$ vertices be given.
- We build $\binom{k}{2}$ DFA’s such that the DFA’s have a non-empty intersection if and only if the graph has a $k$-clique.
- We first need to find $k$ natural numbers $\{n_i\}_{i \in [k]}$ such that:
  - the numbers are at least $n$
  - the numbers are $O(n \log(n))$
  - the numbers are all relatively prime
Reduction

$k$-clique is fpt-reducible to $k$-UDFA-IE.

- The DFA’s read in a unary string that represents a choice of $k$ vertices from $G$.
- The $k$ vertices are encoded by $k$ remainders $\{r_i\}_{i \in [k]}$ where the remainder $r_i$ is relative to $n_i$.
- Each DFA is associated with a pair of numbers from $\{n_i\}_{i \in [k]}$ where the cycle length is the product of these numbers.
- Final states are selected so the DFA only accepts when the vertices associated with the remainders have an edge in $G$. 
Question

Can we solve $k$-UDFA-IE in less than $n^k$ time?

- Yes, there exist faster algorithms for $k$-UDFA-IE [Oliveira, Wehar 2018]
  - 2-UDFA-IE is solvable in $O(n)$ time.
  - 3-UDFA-IE is solvable in $O(n^{1.1865})$ time.
- The faster algorithm for 3-UDFA-IE comes from an equivalence between Triangle Finding and 3-UDFA-IE:
  - For every $\alpha \geq 2$, 3-UDFA-IE is solvable in $O(n^{\frac{\alpha}{2}})$ time if and only if Triangle Finding is solvable in $O(n^\alpha)$ time.
Overview

1. DFA Intersection Problem
2. Complexity of DFA-IE
3. Unary DFA Intersection Problem
4. Complexity of UDFA-IE
5. Open Problems
<table>
<thead>
<tr>
<th>Type of Automata</th>
<th>Bounded Alphabet</th>
<th>Unbounded Alphabet</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFA’s</td>
<td>XNL-complete</td>
<td>XNL-complete</td>
</tr>
<tr>
<td>Group DFA’s</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Commutative DFA’s</td>
<td>$W[1]$-complete</td>
<td>?</td>
</tr>
<tr>
<td>Cycle DFA’s</td>
<td>$W[1]$-complete</td>
<td>?</td>
</tr>
<tr>
<td>Unary DFA’s</td>
<td>$W[1]$-complete</td>
<td>-</td>
</tr>
</tbody>
</table>