Prime Languages[☆,☆☆]

Orna Kupferman, Jonathan Mosheiff

School of Engineering and Computer Science
The Hebrew University, Jerusalem, Israel

Abstract

We say that a deterministic finite automaton (DFA) $\mathcal{A}$ is *composite* if there are DFAs $\mathcal{A}_1, \ldots, \mathcal{A}_t$ such that $L(\mathcal{A}) = \bigcap_{i=1}^{t} L(\mathcal{A}_i)$ and the index of every $\mathcal{A}_i$ is strictly smaller than the index of $\mathcal{A}$. Otherwise, $\mathcal{A}$ is *prime*.

We study the problem of deciding whether a given DFA is composite, the number of DFAs required in a decomposition, decompositions that are based on abstractions, methods to prove primality, and structural properties of DFAs that make the problem simpler or are retained in a decomposition. We also provide an algebraic view of the problem and demonstrate its usefulness for the special case of permutation DFAs.

Keywords: Deterministic Finite Automata (DFA), Regular Languages, DFA Decomposition, Prime DFA, Prime Regular Languages

1. Introduction

Compositionality is a well motivated and studied notion in computer science [2]. By decomposing a problem into several smaller problems, it is possible not only to increase parallelism, but also to sometimes handle inputs that are otherwise intractable. A major challenge is to identify problems and instances that can be decomposed.

Consider for example the LTL model-checking problem [11]. Given a system $\mathcal{S}$ and a specification ψ, checking whether all the computations of $\mathcal{S}$ satisfy ψ can be done in time linear in $\mathcal{S}$ and exponential in ψ. If ψ is a conjunction of smaller specifications, say $\psi = \varphi_1 \wedge \cdots \wedge \varphi_t$, then it is possible to check instead whether $\mathcal{S}$ satisfies each of the φ_i's.[1] Not all problems allow for easy decomposition. For example, if we wish to synthesize a transducer that realizes the specification ψ above, it is not clear how to use

[☆] A preliminary version appeared in the 38th International Symposium on Mathematical Foundations of Computer Science.

[☆☆] The work is supported in part by ERC project no. 278410 – QUALITY.

Email addresses: orna@cs.huji.ac.il (Orna Kupferman), yonatanm@cs.huji.ac.il (Jonathan Mosheiff)

[1] The ability to decompose the specification causes the systems, which are much bigger than the sub-specifications, to be the computational bottleneck in the model-checking problem. Thus, a different big challenge is to decompose the system [12].

the decomposition of ψ into its conjuncts. In particular, it is not clear how to compose a transducer that realizes ψ from t transducers that realize $\varphi_1, \ldots, \varphi_t$ [8].

In the automata-theoretic approach to formal verification, we use automata in order to model systems and their specifications. A natural question then is whether we can decompose a given automaton $\mathcal{A}$ into smaller automata $\mathcal{A}_1, \ldots, \mathcal{A}_t$ such that $L(\mathcal{A}) = \bigcap_{i=1}^{t} L(\mathcal{A}_i)$. Then, for example, we can reduce checking $L(\mathcal{S}) \subseteq L(\mathcal{A})$ to checking whether $L(\mathcal{S}) \subseteq L(\mathcal{A}_i)$ for all $1 \leq i \leq t$.

The automata used for reasoning about systems and their specifications are typically nondeterministic automata on infinite words [13]. As it turns out, however, the question of automata decomposition is open already for the basic model of deterministic automata on finite words (DFAs). Studying DFAs also suggests a very clean mathematical approach, as each regular language has a canonical minimal DFA recognizing it. Researchers have developed a helpful algebraic approach for DFAs that offers some very interesting results on DFAs and their decomposition. To the best of our knowledge, however, the basic question of decomposing a DFA into smaller DFAs is still open. In the algebraic approach, a DFA $\mathcal{A}$ is matched with a *monoid*. [2] The members of this monoid are the actions of words in Σ^* on the states of $\mathcal{A}$. That is, each member is associated with a word and corresponds to the states-to-states transition function induced by the word. In particular, ϵ corresponds to the identity element. The product of two such transition functions is their composition. A DFA is called a *permutation DFA* if its monoid consists of permutations. A DFA is called a *reset DFA* if its monoid consists of constant functions and the identity function. A *wreath product* of a sequence $\mathcal{A}_1, \mathcal{A}_2 \ldots, \mathcal{A}_t$ of DFAs is a cascade in which the transition function of each DFA $\mathcal{A}_i$ may depend on the state of $\mathcal{A}_i$ as well as the states of the DFAs preceding $\mathcal{A}_i$ in the sequence.

The algebraic approach is used in [7] in order to show that every DFA $\mathcal{A}$ can be presented as a wreath product of reset DFAs and permutation DFAs, whose algebraic structure is simpler than that of $\mathcal{A}$. Specifically, the monoid associated with each of the permutation DFA factors in the wreath product is a group. This group divides a subgroup of the monoid DFA of $\mathcal{A}$.

The algebraic approach is based on a syntactic congruence between words in Σ^*: given a regular language $L \subseteq \Sigma^*$, we have that $x \sim_L y$, for words $x, y \in \Sigma^*$, if for every $w, z \in \Sigma^*$, it holds that $w \cdot x \cdot z \in L$ iff $w \cdot y \cdot z \in L$. Thus, the congruence refers to extensions of words from both right and left. In the context of minimization, which motivates the practical study of decomposition, one is interested in a *right congruence*. There, $x \sim_L y$ iff for all words $z \in \Sigma^*$, we have that $x \cdot z \in L$ iff $y \cdot z \in L$. By the Myhill-Nerode theorem [9, 10], the equivalence classes of $\sim_L$ constitute the state space of a minimal canonical DFA for L. The number of equivalence classes is referred to as the *index* of L. We say that a language $L \subseteq \Sigma^*$ is *composite* if there are languages $L_1, \ldots, L_t$ such that $L = \bigcap_{i=1}^{t} L_t$ and the index of L_i, for all $1 \leq i \leq t$, is strictly

[2] A *semigroup* is a non-empty set S coupled with an associative binary operation $\cdot : S \to S$. A semigroup M is called a *monoid* if there exists an *identity element* $e \in M$ such that for every $x \in M$ it holds that $x \cdot e = e \cdot x = x$. A monoid G is called a *group* if for every $x \in G$ there exists an element $x^{-1} \in G$ such that $x \cdot x^{-1} = x^{-1} \cdot x = e$.

smaller than the index of L. Otherwise, we say that L is *prime*[3]. The definitions apply also to DFAs, referring to the languages they recognize.

For example, for Σ with $|\Sigma| > 1$ and $w \in \Sigma^*$, let $L_w = \{w\}$. Clearly, the index of L_w is $|w| + 2$. We claim that if w contains at least two different letters, then L_w is composite. To see this, we show we can express L_w as the intersection of two DFAs of index at most $|w|+1$. Let σ be some letter in w, and let m be its number of occurrences in w. By the condition on w, we have that $1 \leq m < |w|$. It is easy to see that L_w is the intersection of the language w^*, whose index is $|w| + 1$, and the language of all words in which σ appears exactly m times, whose index is $m + 2 \leq |w| + 1$. On the other hand, if w consists of a single letter, then L_w is prime. One of our goals in this work is to develop techniques for deciding primality.

The decomposition of L_w described above is of *width* 2; that is, it has two factors. The case of decompositions of width 2 was studied in [3], where the question of whether one may need wider decompositions was left open. We answer the question positively; that is, we present a language that does not have a decomposition of width 2 but has one of width 3. We conjecture that there exist composite languages of arbitrarily large width. For compositions of width 2, the question of deciding whether a given DFA is composite is clearly in NP, as one can guess the two factors. In the general case, the only bound we have on the width is exponential, which follows from the bound on the size of the underlying DFAs. This bound suggests an EXPSPACE algorithm for deciding whether a given DFA is composite.

Consider a DFA $\mathcal{A}$. We define the *roof* of $\mathcal{A}$ as the intersection of all languages $L(\mathcal{B})$, where $\mathcal{B}$ is a DFA such that $L(\mathcal{A}) \subseteq L(\mathcal{B})$ and the index of $\mathcal{B}$ is smaller than that of $\mathcal{A}$. Thus, the roof of $\mathcal{A}$ is the minimal (with respect to containment) language that can be defined as an intersection of DFAs whose language contain the language of $\mathcal{A}$ and whose index is smaller than the index of $\mathcal{A}$. Accordingly, $\mathcal{A}$ is composite iff $L(\mathcal{A}) = \mathit{roof}(\mathcal{A})$. We use roofs in order to study primality further. In particular, if $\mathcal{A}$ is prime then there exists a word $w \in \mathit{roof}(\mathcal{A}) \setminus L(\mathcal{A})$. The word w is called a *primality witness* for $\mathcal{A}$. Indeed, $\mathcal{A}$ is prime iff it has a primality witness.

Let us go back to the language L_w from the example above. We wish to prove that when $w = \sigma^n$ for some letter σ, then L_w is prime. Let $l > n$ be a natural number such that for every $p \leq n+1$ it holds that $n \equiv l \mod p$. The existence of l is guaranteed by the Chinese remainder theorem. In Section 2, we prove that σ^l is a primality witness for L_w and conclude that L_w is prime. We use the notion of a primality witness to prove the primality of additional, more involved, families of languages.

We then turn to study structural properties of composite and prime DFAs. Each DFA $\mathcal{A}$ induces a directed graph $G_{\mathcal{A}}$. We study the relation between the structure of $G_{\mathcal{A}}$ and the primality of $\mathcal{A}$. We identify cases where primality (with a short primality witness) is guaranteed. An example to such a case are co-safety languages, whose DFA contains one rejecting strongly connected component from which an accepting sink is reachable, We also study structural properties that can be retained in a decomposition and prove, for example, that a composite strongly connected DFA can be decomposed

[3]We note that a different notion of primality, relative to the concatanation operator rather than to intersection, has been studied in [4].

into strongly connected DFAs.

Recall that a decomposition of a DFA $\mathcal{A}$ consists of DFAs that contain the language of $\mathcal{A}$ and are still of a smaller index. A simple way to get a DFA with the above properties is by merging states of $\mathcal{A}$. A *simple decomposition* of $\mathcal{A}$ is a decomposition in which each of the underlying DFAs is a result of merging states of $\mathcal{A}$. Simple decompositions have also been studied in [5] in the context of sequential machines. It follows from [3] that some DFAs have a decomposition of width 2 yet do not have a simple decomposition. We characterize simple decompositions and show that the problem of deciding whether a given DFA has a simple decomposition is in PTIME.

Finally, we develop an algebraic view of DFA decomposition and primality. As [7], our approach is based on the transition monoid of $\mathcal{A}$. First, we show that once we fix the set of accepting states, the question of primality of a DFA $\mathcal{A}$ depends only on $\mathcal{A}$'s transition monoid, rather than its transition function or alphabet. We then focus on permutation DFAs. Given a permutation DFA $\mathcal{A}$ we construct a new DFA, termed the *monoid DFA*, such that compositionally of $\mathcal{A}$ can be reduced to simple-compositionality of its monoid DFA. Driven by observations about monoid DFAs, we show a PSPACE algorithm for deciding the primality of $\mathcal{A}$. We also show that composite permutation DFAs can be decomposed into permutation DFAs.

2. Preliminaries

A *deterministic finite automaton* (DFA) is a 5-tuple $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$, where Q is a finite set of states, Σ is a finite non-empty alphabet, $\delta : Q \times \Sigma \to Q$ is a transition function, $q_0 \in Q$ is an initial state, and $F \subseteq Q$ is a set of accepting states. For $q \in Q$, we use $\mathcal{A}^q$ to denote the DFA $\mathcal{A}$ with q as the initial state. That is, $\mathcal{A}^q = \langle Q, \Sigma, q, \delta, F\rangle$. We extend δ to words in the expected way, thus $\delta : Q \times \Sigma^* \to Q$ is defined recursively by $\delta(q, \epsilon) = q$ and $\delta(q, w_1 w_2 \cdots w_n) = \delta(\delta(q, w_1 w_2 \cdots w_{n-1}), w_n)$.

The *run* of $\mathcal{A}$ on a word $w = w_1 \dots w_n$ is the sequence of states $s_0, s_1 \dots s_n$ such that $s_0 = q_0$ and for each $1 \leq i \leq n$ it holds that $\delta(s_{i-1}, w_i) = s_i$. Note that $s_n = \delta(q_0, w)$. The DFA $\mathcal{A}$ *accepts* w iff $\delta(q_0, w) \in F$. Otherwise, $\mathcal{A}$ *rejects* w. The set of words accepted by $\mathcal{A}$ is denoted $L(\mathcal{A})$ and is called the *language of* $\mathcal{A}$. We say that $\mathcal{A}$ *recognizes* $L(\mathcal{A})$. A language recognized by some DFA is called a *regular language*.

A DFA $\mathcal{A}$ is *minimal* if every DFA $\mathcal{B}$ that has less states than $\mathcal{A}$ satisfies $L(\mathcal{B}) \neq L(\mathcal{A})$. Every regular language L has a single (up to DFA isomorphism) minimal DFA $\mathcal{A}$ such that $L(\mathcal{A}) = L$. The index of L, denoted $ind(L)$, is the size of the minimal DFA recognizing L.

Consider a language $L \subseteq \Sigma^*$. The *Myhill-Nerode relation* relative to L, denoted $\sim_L$, is a binary relation on Σ^* defined as follows: For $x, y \in \Sigma^*$, we say that $x \sim_L y$ if for every $z \in \Sigma^*$ it holds that $x \cdot z \in \Sigma^*$ iff $y \cdot z \in \Sigma^*$. Note that $\sim_L$ is an equivalence relation. It is known that L is regular iff $\sim_L$ has a finite number of equivalence classes. The number of these equivalence classes is equal to $ind(L)$.

Definition 2.1. [DFA decomposition] Consider a DFA $\mathcal{A}$. For $k \in \mathbb{N}$, we say that $\mathcal{A}$ is *k-decomposable* if there exist DFAs $\mathcal{A}_1, \dots, \mathcal{A}_t$ such that for all $1 \leq i \leq t$ it holds that

$ind(\mathcal{A}_i) \leq k$ and $\bigcap_{i=1}^{t} L(\mathcal{A}_i) = L(\mathcal{A})$. The DFAs are then a *$k$-decomposition of $\mathcal{A}$*. The *depth* of $\mathcal{A}$, denoted $depth(\mathcal{A})$, is the minimal k such that $\mathcal{A}$ is k-decomposable.

Obviously, every DFA $\mathcal{A}$ is $ind(\mathcal{A})$-decomposable. The question is whether a decomposition of $\mathcal{A}$ can involve DFAs of a strictly smaller index. Formally, we have the following.

Definition 2.2. [Composite and Prime DFAs] A DFA $\mathcal{A}$ is *composite* if $depth(\mathcal{A}) < ind(\mathcal{A})$. Otherwise, $\mathcal{A}$ is *prime*.

We identify a regular language with its minimal DFA. Thus, we talk also about a regular language being k-decomposable or composite, referring to its minimal DFA. Similarly, for a DFA $\mathcal{A}$, we refer to $ind(L(\mathcal{A}))$ as $ind(\mathcal{A})$.

Example 2.1. Let $\Sigma = \{a\}$ and let $L_k = (a^k)^*$. We show that if k is not a prime power, then L_k is composite. Clearly, $ind(L_k) = k$. If k is not a prime power, then there exist $2 \leq p, q < k$ such that p and q are coprime and $p \cdot q = k$. It then holds that $L_k = L_p \cap L_q$. Since $ind(L_p) < k$ and $ind(L_q) < k$, it follows that L_k is composite.

In Example 3.1, we will show that the above condition is necessary and sufficient, thus, L_k is prime iff k is a prime power.

Let $\mathcal{A}$ be a DFA. We define:

$$\alpha(\mathcal{A}) = \{\mathcal{B} : \mathcal{B} \text{ is a minimal DFA such that L}(\mathcal{A}) \subseteq L(\mathcal{B}) \text{ and } ind(\mathcal{B}) < ind(\mathcal{A})\}.$$

That is, $\alpha(\mathcal{A})$ is the set of DFAs that contain $\mathcal{A}$ and have an index smaller than the index of $\mathcal{A}$. As with other notations, we also refer to $\alpha(L)$, for a regular language L, and we identify $\alpha(L)$ with $\alpha(\mathcal{A})$, for the minimal DFA $\mathcal{A}$ for L. The *roof* of $\mathcal{A}$ is the intersection of the languages of all DFAs in $\alpha(\mathcal{A})$. Thus, $roof(\mathcal{A}) = \bigcap_{\mathcal{B} \in \alpha(\mathcal{A})} L(\mathcal{B})$. Clearly, $L(\mathcal{A}) \subseteq roof(\mathcal{A})$. Also, if $\mathcal{A}$ is composite, then $\alpha(L)$ is an $(ind(\mathcal{A}) - 1)$-decomposition of $\mathcal{A}$. We thus have the following.

Theorem 2.2. *A DFA $\mathcal{A}$ is prime iff $L(\mathcal{A}) \neq roof(\mathcal{A})$, unless $L(\mathcal{A}) = \Sigma^*$.*

The PRIME-DFA problem is to decide, given a DFA $\mathcal{A}$, whether $\mathcal{A}$ is prime. A more general problem is, given a DFA $\mathcal{A}$, to compute $depth(\mathcal{A})$.

We now prove an upper bound on the complexity of PRIME-DFA. We first need the following lemma.

Lemma 2.3. *Let $\mathcal{A}$ be a DFA and let $n = ind(\mathcal{A})$. Then, $|\alpha(\mathcal{A})| = 2^{O(n \cdot |\Sigma| \cdot \log n)}$ and $ind(roof(\mathcal{A})) \leq 2^{2^{O(n \cdot |\Sigma| \cdot \log n)}}$.*

PROOF. Note that a DFA of size smaller than n has an equivalent DFA of size n. Thus, $|\alpha(\mathcal{A})|$ is at most the number of DFAs of index n. A DFA with a state space Q and alphabet Σ can be characterized by a function $f : Q \times \Sigma \to Q$ describing the transition function, a starting state and an accepting state set. Therefore $|\alpha(L)| \leq n^{n \cdot |\Sigma|} \cdot n \cdot 2^n = 2^{O(n \cdot |\Sigma| \cdot \log n)}$.

A DFA for $roof(\mathcal{A})$ can be built using the product DFA construction. As each DFA in $\alpha(L)$ is of size less than n, the size of $\mathcal{A}'$ is at most $n^{|\alpha(L)|} = 2^{2^{O(n \cdot |\Sigma| \cdot \log n)}}$. This bound is doubly-exponential in n. Since $ind(roof(\mathcal{A}))$ is at most the size of $\mathcal{A}'$, we are done.

Combining Theorem 2.2 and Lemma 2.3, we have the following.

Theorem 2.4. *The* PRIME-DFA *problem is in* EXPSPACE*.*

PROOF. We present a NEXPSPACE algorithm that decides the complementary language. Given a DFA $\mathcal{A}$, our algorithm computes a DFA $\mathcal{A}'$ such that $L(\mathcal{A}') = \mathit{roof}(\mathcal{A})$ using the product DFA construction. The algorithm now checks whether $L(\mathcal{A}') \subseteq L(\mathcal{A})$ and accepts or rejects accordingly. Note that $L(\mathcal{A}') \subseteq L(\mathcal{A})$ iff $\mathcal{A}$ is prime.

Let $n = \mathit{ind}(\mathcal{A})$. From the proof of Lemma 2.3, we have that the size of $\mathcal{A}'$ is at most $= 2^{2^{O(n \cdot |\Sigma| \cdot \log n)}}$. The question of whether $L(\mathcal{A}') \subseteq L(\mathcal{A})$ can be nondeterministically decided in space logarithmic in the number of states in $\mathcal{A}$ and $\mathcal{A}'$. The relevant parts of the transition function of $\mathcal{A}'$ can be computed on-the-fly so that $\mathcal{A}'$ needs not be explicitly computed. Therefore, our algorithm indeed decides the problem in NEXPSPACE.

The lower bound we are able to show for the *PRIME-DFA* problem is much lower:

Theorem 2.5. *The* PRIME-DFA *problem is* NLOGSPACE*-hard.*

PROOF. We show a reduction from the problem of deciding the emptiness of a given DFA, known to be NLOGSPACE-hard [6].

Let $\mathcal{C}$ be a minimal composite DFA with a decomposition $L(\mathcal{C}) = \bigcap_{i=1}^{t} \mathcal{C}_i$, where $\mathit{ind}(\mathcal{C}_i) < \mathit{ind}(\mathcal{C})$. For simplicity, we assume that $\mathcal{C}$ does not have a rejecting sink state. The DFA from Example 3.1 can serve as our $\mathcal{C}$.

Given a DFA $\mathcal{A}$, we construct a DFA $\mathcal{B}$ such that $L(\mathcal{A})$ is empty iff $\mathcal{B}$ is prime. Let Σ be the alphabet of $\mathcal{A}$ and let $\#$ be a letter not in Σ. The DFA $\mathcal{B}$ then accepts the $L(\mathcal{A}) \cdot \# \cdot L(\mathcal{C})$. It is not hard to see that $\mathcal{B}$ can be generated in logarithmic space – we only have to add $\#$ transitions from $\mathcal{A}$'s accepting states to $\mathcal{C}$'s initial state.

We prove that indeed $L(\mathcal{A})$ is empty iff $\mathcal{B}$ is prime. Assume first that $L(\mathcal{A}) = \emptyset$. Then, $L(\mathcal{B}) = L(\mathcal{A}) \cdot \# \cdot L(\mathcal{C}) = \emptyset$, which is indeed a prime language.

For the other direction, assume that $L(\mathcal{A}) \neq \emptyset$. We claim that then, $L(\mathcal{B})$ is composite, with the decomposition $\bigcap_{i=1}^{t} L(\mathcal{A}) \cdot \# \cdot L(\mathcal{C}_i)$. Clearly, $L(\mathcal{A}) \cdot \# \cdot L(\mathcal{C}) = \bigcap_{i=1}^{t} L(\mathcal{A}) \cdot \# \cdot L(\mathcal{C}_i)$, thus it is left to prove that for every $1 \leq i \leq t$, it holds that $\mathit{ind}(L(\mathcal{A}) \cdot \# \cdot L(\mathcal{C}_i)) < \mathit{ind}(L(\mathcal{A}) \cdot \# \cdot L(\mathcal{C}))$. To show this, we distinguish between two cases. If $L(\mathcal{A}) = \Sigma^*$, then

$$\mathit{ind}(L(\mathcal{A}) \cdot \# \cdot L(\mathcal{C}_i)) = \mathit{ind}(\mathcal{C}_i) + 1 < \mathit{ind}(\mathcal{C}) + 1 = \mathit{ind}(L(\mathcal{A}) \cdot \# \cdot L(\mathcal{C})),$$

satisfying the inequality. If $L(\mathcal{A}) \neq \Sigma^*$, then

$$\mathit{ind}(L(\mathcal{A}) \cdot \# \cdot L(\mathcal{C}_i)) = \mathit{ind}(\mathcal{A}) + \mathit{ind}(\mathcal{C}_i) + 1 < \mathit{ind}(\mathcal{A}) + \mathit{ind}(\mathcal{C}) + 1 = \mathit{ind}(L(\mathcal{A}) \cdot \# \cdot L(\mathcal{C})),$$

again satisfying the inequality. Hence, $L(\mathcal{B}) = L(\mathcal{A}) \cdot \# \cdot L(\mathcal{C})$ is composite, and we are done

We note the big gap between the upper bound given in Theorem 2.4 and the lower bound given in Theorem 2.5. Tightening these bounds is currently an open problem.

3. Primality Witnesses

Recall that a minimal DFA $\mathcal{A}$ is prime iff $\mathit{roof}(\mathcal{A}) \not\subseteq L(\mathcal{A})$. We define a *primality witness* for $\mathcal{A}$ as a word in $\mathit{roof}(\mathcal{A}) \setminus L(\mathcal{A})$. Clearly, a DFA $\mathcal{A}$ is prime iff $\mathcal{A}$ has a primality witness.

Let $\mathcal{A}$ be a DFA. By the above, we can prove that $\mathcal{A}$ is prime by pointing to a primality witness for $\mathcal{A}$.

The following example utilizes this approach to introduce a family of prime languages. It complements Example 2.1.

Example 3.1. Let $\Sigma = \{a\}$ and let $L_k = (a^k)^*$. We show that if k is a prime power, then L_k is prime. Let $p, r \in \mathbb{N} \cup \{0\}$ be such that p is a prime and $k = p^r$. Let $w_k = a^{(p+1)p^{r-1}}$. Note that $w_k \notin L_k$. We claim that w_k is a primality witness for L_k, and conclude that L_k is prime.

Clearly, $\mathit{ind}(L_k) = k$. Consider a language $L' \in \alpha(L)$. We show that $w_k \in L'$. Assume by contradiction that $w_k \notin L'$. Let $\mathcal{A}'$ be a DFA for L'. Since $\mathit{ind}(L') < \mathit{ind}(L_k) = k$ and $|w_k| > k$, the rejecting run of $\mathcal{A}'$ on w_k must traverse at least one cycle. Let t be the length of such a cycle, and note that $0 < t < k$. Note that for all $i \geq 0$, we have that $a^{it+(p+1)p^{r-1}}$ is not accepted by $\mathcal{A}'$, and hence, $a^{it+(p+1)p^{r-1}} \notin L'$. On the other hand, since $t \not\equiv 0 \mod k$, there exists $i \geq 0$ such that $it \equiv -p^{r-1} \mod k$. For this value of i, we have $a^{it+(p+1)p^{r-1}} \in L \setminus L'$, and thus, $L \not\subseteq L'$, in contradiction to the assumption that $L' \in \alpha(L)$. Therefore, $w_k \in L$ and we are done.

As another example, consider the language $L_w = \{w\}$. In Section 1 we argued that L_w is prime iff w is over a single letter. We now provide the complete proof.

Example 3.2. For $w \in \Sigma^*$, let $L_w = \{w\}$. Assume first that w is not of the form σ^n. Set $w = w_1 \dots w_n$. Note that $\mathit{ind}(L) = n + 2$. Let $\sigma = w_1$ and let m be the number of times that σ appears in w. Since $w \neq \sigma^n$, we have that $m < n$. Now, set $L_1 = w^*$ and $L_2 = \{z \in \Sigma^* : \text{The letter } \sigma \text{ appears in } z \text{ exactly } m \text{ times}\}$.

Note that $\mathit{ind}(L_1) = n + 1$ and that $\mathit{ind}(L_2) = m + 2 < n + 2$. Therefore $L = L_1 \cap L_2$ is a decomposition of L.

For the other direction, assume that $w = \sigma^n$ for some $\sigma \in \Sigma$ and $n \in \mathbb{N} \cup \{0\}$. We claim that L_w is prime. Let $l > n + 1$ be a natural number such that for every $p \leq n + 1$ it holds that $n \equiv l \mod p$. The existence of l is guaranteed by the Chinese remainder theorem. Now, let $z = \sigma^l$. We claim that z is a primality witness for L_w. Note that $\mathit{ind}(L_w) = n + 2$. Let $K \in \alpha(L)$. Note that $w \in K$ and $\mathit{ind}(K) \leq n + 1$. By pumping, there exists $1 \leq p \leq n + 1$ such that $\sigma^{n+ip} \in K$ for every $i \in \mathbb{N} \cup \{0\}$. Since $n \equiv l \mod p$, there exists an i such that $l = n + ip$. Therefore, $z \in K$, proving that z is indeed a primality witness for L and therefore, that L is prime.

We continue to study primality witnesses and focus on their lengths. For a DFA $\mathcal{A}$, we define $\mathit{min\_witness}(\mathcal{A})$ as follows.

$$\mathit{min\_witness}(\mathcal{A}) = \min\{|w| : w \text{ is a primality witness for } \mathcal{A}\}.$$

Likewise, for a language L with minimal DFA $\mathcal{A}$, we define $\mathit{min\_witness}(L) = \mathit{min\_witness}(\mathcal{A})$.

Proposition 3.3. *A prime DFA has a primality witness of length doubly exponential.*

PROOF. Let $n = ind(\mathcal{A})$ and let $\mathcal{A}'$ be the minimal DFA of $roof(\mathcal{A})$. From Lemma 2.3, we have that $ind(\mathcal{A}') \leq 2^{2^{O(n \cdot |\Sigma| \cdot \log n)}}$. Using the product construction, we define a DFA $\mathcal{A}''$ such that $L(\mathcal{A}'') = roof(\mathcal{A}) \setminus L$. The DFA $\mathcal{A}''$ is a product of $\mathcal{A}$ and $\mathcal{A}'$ and therefore, $ind(\mathcal{A}'') \leq ind(\mathcal{A}) \cdot ind(\mathcal{A}')$. Note that $L(\mathcal{A}'')$ contains exactly all the primality witnesses for $\mathcal{A}$.

Since $\mathcal{A}$ is prime, the language of $\mathcal{A}''$ is not empty. Let $w \in \Sigma^*$ be the shortest word in $L(\mathcal{A}'')$. The word w is the shortest primality witness for $\mathcal{A}$. Since w is the shortest word in $L(\mathcal{A}'')$, the run of $\mathcal{A}''$ on w is simple (that is, has no cycles). Therefore,

$$|w| \leq ind(\mathcal{A}'') \leq ind(\mathcal{A}) \cdot ind(\mathcal{A}'') \leq n \cdot 2^{2^{O(n \cdot |\Sigma| \cdot \log n)}}$$

which is doubly exponential in $ind(\mathcal{A})$, and we are done.

Proposition 3.3 implies a naive algorithm for PRIME-DFA: Given an input DFA $\mathcal{A}$, the algorithm proceeds by going over all words $w \in \Sigma^*$ of length at most $2^{2^{O(n \cdot |\Sigma| \cdot \log n)}}$, and checking, for each $\mathcal{B} \in \alpha(\mathcal{A})$, whether $w \in L(\mathcal{B})$. While the algorithm is naive, it suggests that if we strengthen Proposition 3.3 to give a polynomial bound on $min\_witness(\mathcal{A})$, we would have a PSPACE algorithm for PRIME-DFA. The question of whether such a polynomial bound exists is currently open.

The following examples introduce more involved families of prime languages.

Example 3.4. For $n \in \mathbb{N}$, let $K_n = \{ww : w \in \Sigma^n\}$ and $L_n = comp(K_n^*)$; that is, $L_n = \Sigma^* \setminus K_n^*$. Let w_n be a concatenation of all words of the form ss for $s \in \Sigma^n$ in some arbitrary order. Note that $w_n \notin L_n$. We prove that w_n is a primality witness for L_n, and conclude that L_n is prime. Furthermore, $min\_witness(L_n)$ is polynomial in $ind(L)$.

We first describe the Myhill-Nerode equivalence classes of L_n. These consist of the following types of classes.

1. Classes of the form $K_n^* \cdot \{x\}$ for every $x \in \Sigma^*$ such that $0 \leq |x| < n$. There are $2^n - 1$ such classes.
2. Classes of the form $K_n^* \cdot \{x \cdot y \cdot x : x \in \Sigma^* \text{ and } |x| = n-k\}$ for every $0 < k \leq n$ and $y \in \Sigma^k$. There are $2^{n+1} - 2$ such classes.
3. The single accepting sink class $\{x \in \Sigma^* : \text{for every } y \in \Sigma^* \text{ it holds that } x \cdot y \notin K_n\}$.

Overall, we have $ind(L_n) = 3 \cdot 2^n - 2$. Let $\mathcal{A}_n = \langle Q, \Sigma, q_0, \delta, F \rangle$ be the minimal DFA for L_n. Consider a language $L' \subseteq \Sigma^*$ with $ind(L') < ind(L_n)$ such that $w_n \notin L'$. To prove that w_n is a primality witness for L_n, we need to show that $L_n \not\subseteq L'$. Let $\mathcal{A}' = \langle Q', \Sigma, q_0', \delta', F' \rangle$ be the minimal DFA for L'. It is not hard to see that for every $q \in Q$ other than the accepting sink, there exists a prefix x of w_n such that $\delta(q_0, x) = q$. Since $ind(L') < ind(L_n)$, there exist two prefixes x, y of w_n such that $\delta(q_0, x) \neq \delta(q_0, y)$ but $\delta'(q_0', x) = \delta'(q_0', y)$. Assume without loss of generality that x is shorter than y and let z be such that $w_n = yz$. It then holds that $\delta'(q_0', xz) = \delta'(q_0', w_n) \notin F'$ and therefore $xz \notin L'$. We claim that $xz \in L_n$, implying that $L_n \not\subseteq L'$. In order to verify that $xz \in L_n$, we need to distinguish between 3 cases:

1. The case $|x| \not\equiv |y| \bmod 2n$. Then, $|xz| \not\equiv 0 \bmod 2n$, and therefore $xz \in L_n$.
2. The case $x \equiv y \bmod 2n$ and $x \bmod 2n < n$. Then, $x \in K_n^* \cdot w$ and $y \in K_n^* \cdot w'$ for some $0 \leq k < n$ and $w, w' \in \Sigma^k$ such that $w \neq w'$. Assume that $z = z_1 \cdots z_m$ with $z_i \in \Sigma$. Then it holds that $z_{n-k+1} \cdots z_n = w' \neq w$. Therefore, $xz \in L_n$.
3. The case $x \equiv y \bmod 2n$ and $x \bmod 2n \geq n$. Then, there exist $0 \leq k < n$, $w = w_1 \cdots w_n \in \Sigma^*$m and $w' = w'_1 \ldots w'_n$ such that $x \in K_n^* \cdot w \cdot w_1 \ldots w_k$ and $y \in K_n^* \cdot w' \cdot w'_1 \cdots w'_k$ and such that $w'_{k+1} \cdots w'_n \neq w_{k+1} \cdots w_n$. Assume that $z = z_1 \cdots z_m$ with $z_i \in \Sigma$. Then, it holds that $z_1 \cdots z_{n-k} = w'_{k+1} \cdots w'_n \neq w_{k+1} \cdots w_n$. Therefore, $xz \in L_n$.

We have shown that $L_n \not\subseteq L'$, implying that w_n is indeed a primality witness for L_n. Note that $|w_n| = 2^{n+1} \cdot n = O(ind(L_n) \cdot \log(ind(L_n)))$, proving that L_n has primality witnesses of length polynomial in $ind(L_n)$.

Example 3.5. Consider words $s = s_1 \cdots s_m$ and $w = w_1 \cdots w_t$, both over the same alphabet Σ. If there exists an increasing sequence of indices $1 \leq i_1 < i_2 < \cdot < i_t \leq m$ such that for each $1 \leq j \leq t$ it holds that $w_j = s_{i_j}$, we say that w is a *subsequence* of s. If w is a subsequence of s and $w \neq s$, then we say that w is a *proper subsequence* of s.

Let $w \in \Sigma^*$. Consider the language $L_w = \{s \in \Sigma^* : w \text{ is a subsequence of } s\}$. We claim that L_w is prime, and furthermore, that $min\_witness(L_w) \leq 2 \cdot ind(L_w)$.

If $w = \epsilon$ then $L_w = \Sigma^*$ and the claim is trivial. Thus, we may assume $w \neq \epsilon$.

Let $n = |w|$ and $w = w_1 \cdots w_n$. Note that $ind(L_w) = n + 1$. The Myhill-Nerode equivalence classes defined by the language L_w are as follows:

- For every $0 \leq k < n$, the set $\{s \in \Sigma^* : w_1 \cdots w_k \text{ is a subsequence of } s \text{ but } w_1 \cdots w_{k+1} \text{ is not a subsequence of } s\}$ is an equivalence class. There are n such sets.
- The set L_w itself is an equivalence class.

We first prove the following claim.

Claim 3.6. *Let $w = w_1 \ldots w_n \in \Sigma^*$ with $n > 0$. There exists a word $s \in \Sigma^*$ such that the following hold:*

1. *w is not a subsequence of s.*
2. *For every word $c \in \Sigma^*$ such that $|c| < n$ and c is a subsequence of w, it holds that c is a subsequence of s.*

Furthermore, there exists such a word s for which $|s| < 2|w|$.

PROOF. For each $1 \leq i \leq n-1$, we define x_i and y_i as follows:

- $x_i = w_{i+1}$.
- $y_i = \begin{cases} \epsilon & \text{if } w_i = w_{i+1}, \\ w_i & \text{if } w_i \neq w_{i+1}. \end{cases}$

Define $s = x_1 \cdot y_1 \cdot x_2 \cdot y_2 \cdots x_{n-1} \cdot y_{n-1}$. We show that s satisfies the requirements in the lemma. We claim that for every i, the longest prefix of w that is a subsequence of $x_1 \cdot y_1 \cdot x_2 \cdot y_2 \cdots x_i \cdot y_i$ is the word $w_1 \cdots w_i$. This claim can be easily verified by an induction on i. Symmetrically, we claim that for every i, the longest suffix of w that is a subsequence of $x_i.y_i \ldots x_{n-1}.y_{n-1}$ is $w_{i+1} \ldots w_n$. This claim can be verified by an induction on $n - i$.

From the first claim, the longest prefix of w that is a subsequence of s is $w_1 \ldots w_{n-1}$. Therefore, w is not a subsequence of s, implying that s satisfies the first requirement.

Let $c \in \Sigma^{n-1}$ be such that c is a subsequence of w. To show that s satisfies the second requirement, it is enough to show that c is a subsequence of s. Assume that c is of the form $c = w_1 \cdots w_{i-1} \cdot w_{i+1} \cdots w_n$ for some i. From the above claims about s, we know that $w_1 \ldots w_{i-1}$ is a subsequence of $x_1 y_1 \ldots x_{i-1} y_{i-1}$ and that $w_{i+1} \ldots w_n$ is a subsequence of $x_i \cdot y_i \cdots x_n \cdot y_n$. Therefore, c is a subsequence of s.

We have shown that s satisfies both requirements. Finally, clearly, $|s| < 2n$.

We now continue to complete the claim on the primality of L_w. Let $w = w_1 \ldots w_n \in \Sigma^*$ and let s be the string guaranteed by Claim 3.6. We claim that s is a primality witness for L_w.

Let $L \in \alpha(L_w)$ and let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F \rangle$ be a minimal DFA for L. Let $c \in \Sigma^*$ be a proper subsequence of w. Note that $s \in L_c$. Therefore, to show that s is a primality witness for L_w, it is enough to show that there exists a proper subsequence c of w such that $L_c \subseteq L$.

For each $0 \leq i \leq n$, let $Q_i = \{q \in Q : L(\mathcal{A}^q) \subseteq L_{w_{i+1} \cdots w_n}\}$. Note that $L_w = \Sigma^* \cdot L_w$. Hence, every reachable state in $\mathcal{A}$ is in Q_0. Since $\mathcal{A}$ is minimal, all states are reachable and so $Q = Q_0$. Note that for every $1 \leq i \leq n$, and for every word x that contains the letter w_i, it holds that $\delta(Q_{i-1}, x) \subseteq Q_i$.

Now, for every $0 \leq i < n$ let $R_i = Q_i \setminus Q_{i+1}$ and $R_n = Q_n$. The sets $R_0 \ldots R_n$ are a partition of Q into $n + 1$ disjoint sets. Since $\mathcal{A}$ has at most n states, one of these sets must be empty. It is impossible to have $R_n = Q_n = \emptyset$ since $\delta(q_0, w) \in R_n$. Therefore, there exists $0 \leq i < n$ such that $R_i = \emptyset$. Set $c = w_1 \ldots w_i w_{i+2} \ldots w_n$. We now claim that $L_c \subseteq L$. Let $x \in L_c$. We can write x as a concatenation of words $x = x_1 \cdot x_2 \cdots x_i \cdot x_{i+2} \cdots x_n$ such that for each $j \neq i + 1$, the word x_j contains the letter w_i. Set $q = \delta(q_0, x_1 \cdot x_2 \cdots x_i)$. It holds that $q \in Q_i$, but since $R_i = \emptyset$, this implies $q \in Q_{i+1}$. Therefore, $L(\mathcal{A}^q) \subseteq L_{w_{i+2} \ldots w_n}$, so $x_{i+2} \ldots x_n \in L(\mathcal{A}^q)$. Hence, $x \in L(\mathcal{A}) = L$ as required. We have shown that $L \subseteq L_c$, which proves that s is indeed a witness for the primality of L_w. Furthermore note that s is of length at most $2n + 1 < 2 \cdot ind(L_w)$.

4. The Width of a Decomposition

Languages that can be decomposed into two factors have been studied in [3], where the question of whether one may need more than two factors was left open. In this section we answer the question positively. Formally, we have the following. Let $\mathcal{A}$ be a DFA. If there exist DFAs $\mathcal{A}_1, \mathcal{A}_2 \ldots \mathcal{A}_m \in \alpha(\mathcal{A})$ such that $\mathcal{A} = \bigcap_{i=1}^{m} L(\mathcal{A}_i)$, we say that $\mathcal{A}$ is *m-factors composite*. Assume that $\mathcal{A}$ is composite. Then, $width(L)$ is defined as the minimal m such that $\mathcal{A}$ is m-factors composite. Clearly, for every composite $\mathcal{A}$,

it holds that $width(\mathcal{A}) \geq 2$. The question left open in [3] is whether there exists a composite $\mathcal{A}$ such that $width(L) > 2$. Such a language is presented in the following example.

Example 4.1. Let $\Sigma = \{a, b, c\}$ and let $L = \{w \in \Sigma^* : w$ is a prefix of a word in $c^*.(a^+.b^+.c^+)^*\}$. We show that the language L is composite with $width(L) = 3$. Consider $\mathcal{A}$, the minimal DFA for L, described in Figure 1. A decomposition of $\mathcal{A}$ is given by the DFAs described in Figure 2. Note that all three factors are of index 3, while $\mathcal{A}$ is of index 4. Missing edges lead to a rejecting sink, hence the additional state in the index.

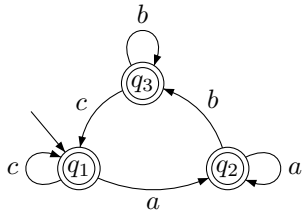

Figure 1: The DFA $\mathcal{A}$ is of width 3.

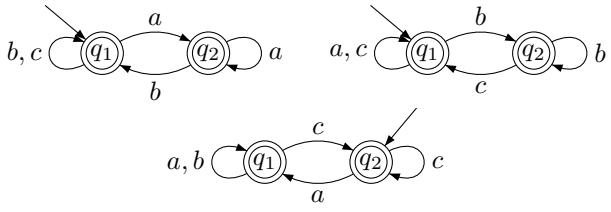

Figure 2: A decomposition for $\mathcal{A}$.

It can be verified by a case-by-case analysis that L is not 2-factor composite.

Example 4.1 motivates us to conjecture that the width of composite languages is unbounded. That is, that width induces a strong hierarchy on the set of composite languages.

On the other hand, given a composite $L \subseteq \Sigma^*$, we wish to provide an upper bound on $width(L)$. We conjecture that there exists a polynomial f such that $width(L) \leq f(ind(L))$ for some polynomial f. If this is true, the algorithm given in the proof of Theorem 2.4 can be improved to a PSPACE algorithm by going over all subsets of $D \subseteq \alpha(\mathcal{A})$ such that $|D| \leq f(ind(\mathcal{A}))$, and checking for each such D whether $\bigcap_{\mathcal{B} \in D} L(\mathcal{B}) = L(\mathcal{A})$.

5. Structural Properties

Consider a minimal DFA $\mathcal{A}$ and a DFA $\mathcal{B} \in \alpha(\mathcal{A})$. Recall that $L(\mathcal{A}) \subseteq L(\mathcal{B})$ and $ind(\mathcal{B}) < ind(\mathcal{A})$. Thus, intuitively, in $\mathcal{B}$, fewer states have to accept more words. In this section we examine whether this requirement on $\mathcal{B}$ can be of help in reasoning about possible decompositions.

The DFA $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ induces a directed graph $G_{\mathcal{A}} = \langle Q, E\rangle$, where $E = \{(q, q') : \exists \sigma \in \Sigma \text{ such that } \delta(q, \sigma) = q'\}$. The strongly connected components (SCCs) of $G_{\mathcal{A}}$ are called the SCCs of $\mathcal{A}$. We refer to the directed acyclic graph (DAG) induced by the SCCs of $G_{\mathcal{A}}$ as the SCC DAG of $\mathcal{A}$. A *leaf* in $G_{\mathcal{A}}$ is a SCC that is a sink in this DAG. A DFA $\mathcal{A}$ is said to be *strongly connected* if it consists of a single SCC.

Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ and $\mathcal{B} = \langle S, \Sigma, s_0, \eta, G\rangle$ be DFAs. Let $q \in Q$ and $s \in S$. If there exists a word $w \in \Sigma^*$ such that $\delta(q_0, w) = q$ and $\eta(s_0, w) = s$, then we say that q *touches* s, denoted $q \sim s$. Obviously, this is a symmetric relation.

Lemma 5.1. *Let $\mathcal{A}$ and $\mathcal{B}$ be DFAs such that $L(\mathcal{A}) \subseteq L(\mathcal{B})$ and let q and s be states of $\mathcal{A}$ and $\mathcal{B}$, respectively, such that $q \sim s$. Then, $L(\mathcal{A}^q) \subseteq L(\mathcal{B}^s)$.*

PROOF. Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ and $\mathcal{B} = \langle S, \Sigma, s_0, \eta, G\rangle$ be DFAs. Let $w \in \Sigma^*$ be such that $\delta(q_0, w) = q$ and $\eta(s_0, w) = s$. Let $x \in L(\mathcal{A}^q)$. Since $w \cdot x \in L(\mathcal{A})$ and $L(\mathcal{A}) \subseteq L(\mathcal{B})$, we have that $w \cdot x \in L(\mathcal{B})$, and thus, $x \in L(\mathcal{B}^s)$. Hence, $L(\mathcal{A}^q) \subseteq L(\mathcal{B}^q)$.

For each $s \in S$, consider the subset of Q consisting of the states that touch s. Recall that $|S| < |Q|$. Intuitively, if one attempts to design $\mathcal{B}$ so that $L(\mathcal{B})$ over-approximates $L(\mathcal{A})$ as tightly as possible, one would try to avoid, as much as possible, having states in S that touch more than one state in Q. However, by the pigeonhole principle, there must be a state $s \in S$ that touches more than one state in Q. The following lemma provides a stronger statement: There must exist a non-empty set $Q' \subseteq Q$ relative to which the DFA $\mathcal{B}$ is "confused" when attempting to imitate $\mathcal{A}$.

Lemma 5.2. *Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ and $\mathcal{B} = \langle S, \Sigma, s_0, \eta, G\rangle$ be minimal DFAs such that $\mathcal{B} \in \alpha(\mathcal{A})$. Then, there exists a non empty set $Q' \subseteq Q$ such that for every $q_1 \in Q'$ and $s \in S$ with $q_1 \sim s$, there exists $q_2 \in Q'$ such that $q_1 \neq q_2$ and $q_2 \sim s$.*

PROOF. Since $\mathcal{A}$ and $\mathcal{B}$ are minimal, all states in Q and S are reachable. Therefore, every state in Q touches some state in S. We now construct two sequences $Q_0, Q_1, Q_2 \ldots$ and $S_0, S_1, S_2 \ldots$ as follows.

First, $Q_0 = Q$ and $S_0 = S$. Now, if there exist $q \in Q_i$ and $s \in S_i$, for $i \geq 0$, such that s touches q and s touches no other state in Q_i, we set $Q_{i+1} = Q_i \setminus \{q\}$ and $S_{i+1} = S_i \setminus \{s\}$. We repeat this step until no such q and s exist. If no such q and s exist, we are done.

This process eventually terminates, since the sequence $|Q_i|$ is decreasing. Assume that the process terminates after m iterations. Set $Q' = Q_m$. We claim that Q' satisfies the lemma's conditions. First, note that since $|S| < |Q|$, then for every $i >= 0$ it holds that $|S_i| < |Q_i|$ and therefore $Q' \neq \emptyset$.

Now, let $q_1 \in Q'$ and let $s \in S$ such that s touches q_1. If there exists i such that $s \in S_i \setminus S_{i+1}$, then there exists $q \in Q_i \setminus Q_{i+1}$ such that s touches q but doesn't touch any other member of Q_i. However, $q_1 \in Q_i$ is also touched by s, leading to contradiction. Therefore, $s \in S_m$, and since the process halts after m iterations, there must exist $q_2 \in Q'$ such that s touches q_2. This proves that Q' satisfies the lemma's conditions.

We are going to use Lemma 5.2 in our study of primality of classes of DFAs. We start with safe and co-safe DFAs.

A language $L \subseteq \Sigma^*$ is a *safety* language if for every $w \notin L$ there exists a prefix x of w such that $x \cdot y \notin L$ for all $y \in \Sigma^*$. A language $L \subseteq \Sigma^*$ is a *co-safety* language if $comp(L)$ is a safety language. That is, L is *co-safety* if for every $w \in \Sigma^*$ there exists a prefix x of w such that $x \cdot y \in L$ for all $y \in \Sigma^*$. The word x is called a *good prefix* of L [1].

Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F \rangle$ be a minimal DFA such that $L(\mathcal{A}) \neq \Sigma^*$. It is easy to see that $L(\mathcal{A})$ is *co-safety* iff $\mathcal{A}$ has a single accepting state s, which is an accepting sink. Obviously, the singleton $\{s\}$ is a SCC of $\mathcal{A}$. If $Q \setminus \{s\}$ is a SCC, we say that $\mathcal{A}$ is a *simple co-safety* DFA. For example, it is not hard to see that for all $w \in \Sigma^*$, if $|\Sigma| > 2$, then the language L_w of all words that have w as a subword is such that the minimal DFA for L_w is a simple co-safety DFA.

Example 5.3. Fix Σ such that $|\Sigma| \geq 3$. For $w \in \Sigma^*$, let $L_w = \{v \in \Sigma^* : w \text{ is a substring of } v\}$. It is easy to verify that for all $w \in \Sigma^*$, the minimal DFA for L_w is a simple co-safety DFA.

We prove that simple co-safe DFAs are prime, and they have short primality witnesses. We first need the following lemma.

Lemma 5.4. *Let $\mathcal{A}$ be a simple co-safe DFA of index k. For every word $x \in \Sigma^*$ such that $x \notin L(\mathcal{A})$ and DFA $\mathcal{B} \in \alpha(\mathcal{A})$, there exists a word $y_\mathcal{B} \in \Sigma^*$ such that $|y_\mathcal{B}| < k^2+k$, $x \cdot y_\mathcal{B} \notin L(\mathcal{A})$, and $x \cdot y_\mathcal{B} \cdot \Sigma^* \subseteq L(\mathcal{B})$. Furthermore, given $x \in \Sigma^*$, it is possible to choose the words $y_\mathcal{B}$ for x and for each $\mathcal{B} \in \alpha(\mathcal{A})$ so that $|\{y_\mathcal{B} : \mathcal{B} \in \alpha(\mathcal{A})\}| \leq k^2$.*

PROOF. Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F \rangle$. Denote the two SCCs of $\mathcal{A}$ by C_1 and C_2, where C_2 is the singleton accepting sink. Assume that $\mathcal{B} = \langle S, \Sigma, s_0, \eta, G \rangle$. Let $w \in L(\mathcal{A})$. Note that for every $y \in \Sigma^*$ it holds that $w \cdot y \in L(\mathcal{A})$. Since $L(\mathcal{A}) \subseteq L(\mathcal{B})$, it also holds that $w \cdot y \in L(\mathcal{B})$. Therefore, the state $\eta(s_0, w)$ is an accepting sink of $\mathcal{B}$. Since $\mathcal{B}$ is minimal, it only has a single accepting sink. Let $\overline{s} \in S$ be this accepting sink.

Let $Q' \subseteq Q$ be the set of states of $\mathcal{A}$ guaranteed by Lemma 5.2, with respect to $\mathcal{B}$. Note that $|Q'| \geq 2$. Let $q' \in Q'$ be such that the set $L(\mathcal{A}^{q'})$ is minimal relative to the containment partial order. For every $q'' \neq q' \in Q'$ it holds that $L(\mathcal{A}^{q''}) \not\subseteq L(\mathcal{A}^{q'})$. Obviously, $q' \in C_1$.

Given x, let $q = \delta(q_0, x)$. Since $x \notin L(\mathcal{A})$, we have $q \in C_1$. Since $q' \in C_1$, there exists a word $w \in \Sigma^*$ such that $\delta(q, w) = q'$ and $|w| \leq k - 2 = |C_1| - 1$. Let $s = \eta(s_0, x \cdot w)$. Note that $s \sim q'$, and therefore, there exists a state $q'' \in Q'$ such that $q'' \neq q'$ and $s \sim q''$. Furthermore, there exists $z \in L(\mathcal{A}^{q''}) \setminus L(\mathcal{A}^{q'})$. Let $y_\mathcal{B} = wz$ and we have $\delta(q_0, x \cdot y_\mathcal{B}) = \delta(q_0, x \cdot w \cdot z) = \delta(q, w \cdot z) = \delta(q', z) \notin F$.

Therefore, $x \cdot y_{\mathcal{B}} \notin L(\mathcal{A})$. Let $x' \in \Sigma^*$ be a word such that $\delta(q_0, x') = q''$ and $\eta(s_0, x') = s$. Note that $\delta(q_0, x' \cdot z) = \delta(q'', z) \in F$. Therefore, $x' \cdot z \in L(\mathcal{A})$. It follows that $\eta(s_0, x \cdot y) = \eta(s_0, x \cdot w \cdot z) = \eta(s, z) = \eta(s, x' \cdot z) = \overline{s}$. Therefore, $x \cdot y_{\mathcal{B}} \cdot \Sigma^* \subseteq L(\mathcal{B})$ as required.

We now show that there exists $z \in L(\mathcal{A}^{q''}) \setminus L(\mathcal{A}^{q'})$ such that $|z| < k^2$, and conclude that $|y_{\mathcal{B}}| = |w| + |z| < k^2 + k$. Let z be a string of minimal length in $L(\mathcal{A}^{q''}) \setminus L(\mathcal{A}^{q'})$. Let z = $z_1 \cdots z_t$. Assume by way of contradiction that $t \geq k^2$. For each $0 \leq i \leq t$, define $a_i = \delta(q', z_1 \cdots z_i)$ and $b_i = \delta(q'', z_1 \ldots z_i)$. For every i, we have that $(a_i, b_i) \in Q^2$ and $|Q^2| = k^2$. Since $t \geq k^2$, there exist $0 \leq i < j \leq t$ such that $(a_i, b_i) = (a_j, b_j)$, but then we have $z_1 \cdots z_i \cdot z_{j+1} \cdots z_t \in L(\mathcal{A}^{s'}) \setminus L(\mathcal{A}^s)$, contradicting the minimality of $|z|$. Therefore, $|z| < k^2$ and $|y_{\mathcal{B}}| < k^2 + k$.

Note that $y_{\mathcal{B}}$ is only dependent on q' and q''. Therefore, by making $y_{\mathcal{B}}$ a function of q' and q'', we have $|\{y_{\mathcal{B}} : A' \in \alpha(\mathcal{A})\}| \leq |Q \times Q| = k^2$.

Theorem 5.5. *Every simple co-safe DFA is prime with a primality witness of polynomial length.*

PROOF. Given a simple co-safe DFA $\mathcal{A}$, we define the primality witness for $L(\mathcal{A})$ as $w = y_0 \cdot y_1 \cdots y_t$, where $y_0, y_1, \ldots, y_t$ is the finite sequence of words over Σ^* defined by the following algorithm:

1. Let $D_0 = \alpha(\mathcal{A})$.
2. Let $y_0 = \epsilon$.
3. To define y_{i+1}, set $x = y_0 \cdot y_1 \cdots y_i$. Let $\mathcal{B} \in D_i$. By Lemma 5.4, there exists a word $y_{\mathcal{B}} \in \Sigma^*$ such that $x \cdot y_{\mathcal{B}} \notin L(\mathcal{A})$ and $x \cdot y_{\mathcal{B}} \cdot \Sigma^* \subseteq L(\mathcal{B})$. Furthermore, it is possible to choose the words $y_{\mathcal{B}}$ such that $|\{y_{\mathcal{B}} : \mathcal{B} \in \alpha(\mathcal{A})\}| \leq k^2$. Therefore, there exists a subset $D_i' \subseteq D_i$ such that $|D_i'| \geq \frac{|D_i|}{k^2}$ and $y_{\mathcal{B}}$ is identical for every $\mathcal{B} \in D_i'$. Let y_{i+1} be this $y_{\mathcal{B}}$, and let $D_{i+1} = D_i \setminus D_i'$.
4. Repeat Step 3, increasing i, until $D_i = \emptyset$.

We claim that w is a primality witness for $L(\mathcal{A})$. Let $\mathcal{B} \in \alpha(L)$. There exists $0 \leq i \leq t$ such that $\mathcal{B} \in D_i'$. Note that $y_{i+1} = y_{\mathcal{B}}$. Therefore, $y_0 \cdot y_1 \cdots y_{i+1} \cdot \Sigma^* \subseteq L(\mathcal{B})$. Since $w \in y_0 \cdot y_1 \cdots y_{i+1} \cdot \Sigma^*$, we have $w \in L(\mathcal{B})$. On the other hand, $w \notin L(\mathcal{A})$, making it a primality witness.

Now, note that for every i it holds that $\frac{|D_{i+1}|}{|D_i|} \leq \frac{k^2-1}{k^2}$. Therefore,

$$(\frac{k^2}{k^2-1})^t \leq |\alpha(\mathcal{A})| \leq 2^{O(k \cdot \log k \cdot |\Sigma|)}.$$

Hence, t is bounded by a polynomial in k. Now, note that for every i it holds that $|y_i| \leq k^2 + k$. Therefore, $|w| \leq t \cdot (k^2 + k)$ is also bounded by a polynomial in k.

The requirement on $\mathcal{A}$ being simple is essential. For example, consider the DFA $\mathcal{A}$ appearing in Figure 3. While $\mathcal{A}$ is co-safe, it is not simple: It has two SCCs C_1 and C_2 such that no path exists either from C_1 to C_2 or from C_2 to C_1. In such a case, $\mathcal{A}$ can be decomposed into DFAs $\mathcal{A}_1$ and $\mathcal{A}_2$ as in the example in Figure 3. On the other hand, consider a non simple co-safety DFA in which the SCCs are linearly ordered by a path that traverses all of them. In this case, the question of primality remains open.

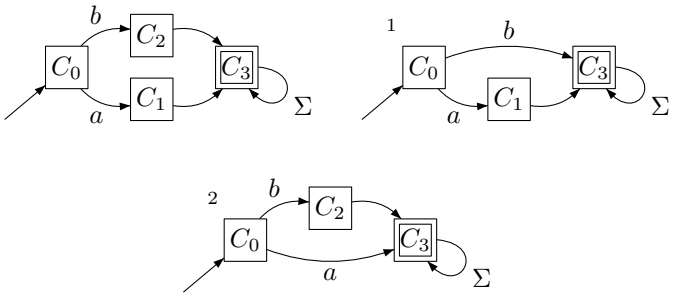

Figure 3: The DFA $\mathcal{A}$ can be decomposed into $\mathcal{A}_1$ and $\mathcal{A}_2$.

Our main result about the structural properties of composite DFAs shows that strong connectivity can be carried over to the DFAs in the decomposition. Intuitively, let $\mathcal{A}_i$ be a member of a decomposition of $\mathcal{A}$, and let $\mathcal{B}_i$ be a DFA induced by a leaf of the SCC graph of $\mathcal{A}_i$. We can replace $\mathcal{A}_i$ by $\mathcal{B}_i$ and still get a valid decomposition of $\mathcal{A}$. Formally, we have the following.

Theorem 5.6. *Let $\mathcal{A}$ be a strongly connected composite DFA. Then, $\mathcal{A}$ can be decomposed using only strongly connected DFAs as factors.*

PROOF. Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$. Since $\mathcal{A}$ is composite, there exist $\mathcal{A}_1 \dots \mathcal{A}_m \in \alpha(\mathcal{A})$ such that $L(\mathcal{A}) = \bigcap_{i=1}^{m} L(\mathcal{A}_i)$. Let $\mathcal{A}_i = \langle Q_i, \Sigma, q_0^i, \delta_i, F_i\rangle$. We recursively define a sequence of words $w_1 \dots w_m$ as follows. Let $q^i = \delta_i(q_0^i, w_1 \cdots w_{i-1})$. Consider the SCC DAG of $\mathcal{A}_i$. Let w_i be a word such that the state $\delta_i(q^i, w_i)$ is in a leaf of the SCC DAG.

Now, let $q = \delta(q_0, w_1 \cdots w_m)$, and let w' be a string that satisfies $\delta(q, w') = q_0$. Such a string exists since $\mathcal{A}$ is strongly connected. Finally, define $w = w_1 \cdot w_2 \cdots w_m \cdot w'$. Note that $\delta(q_0, w) = q_0$.

Let $1 \leq i \leq m$. Note that the run of $\mathcal{A}_i$ on the string w, ends inside a leaf of the SCC DAG of $\mathcal{A}_i$. Let $\mathcal{B}_i$ be a DFA consisting only of the states of that leaf, with the initial state $\delta_i(q_0^i, w)$. Note that $\mathcal{B}_i$ is strongly connected, and that $ind(\mathcal{B}_i) \leq ind(\mathcal{A}_i)$. We claim that $L(\mathcal{A}) = \bigcap_{i=1}^{m} L(\mathcal{B}_i)$, showing that $\mathcal{A}$ can be t-decomposed into strongly connected DFAs.

Let $z \in \Sigma^*$. Recall that $\delta(q_0, w) = q_0$. Therefore, $z \in L(\mathcal{A})$ iff $w \cdot z \in L(\mathcal{A})$. Recall that the initial state of $\mathcal{B}_i$ is $\delta_i(q_0^i, w)$. Therefore, for every $1 \leq i \leq m$, it holds that $w \cdot z \in L(\mathcal{A}_i)$ iff $z \in L(\mathcal{B}_i)$. We conclude that $z \in L(\mathcal{A})$ iff $z \in \bigcap_{i=1}^{m} L(\mathcal{B}^i)$, and we are done.

We find the result surprising, as strong connectivity significantly restricts the over-approximating DFAs in the decomposition.

6. Simple Decompositions

Consider the task of decomposing a DFA $\mathcal{A}$. A natural approach is to build the factors of $\mathcal{A}$ by merging states of $\mathcal{A}$ into equivalence classes in such a manner that the transition function of $\mathcal{A}$ respects the partition of its states into equivalence classes. [4] The result of such a construction is a DFA $\mathcal{B}$ that contains $\mathcal{A}$ and still has fewer states. If $\mathcal{A}$ has a decomposition into factors constructed by this approach, we say that $\mathcal{A}$ is *simply-composite*. In this section, we formally define the concept of a simple decomposition and investigate its properties. In particular, we show that it is computationally easy to check whether a given DFA is simply-composite.

Consider DFAs $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ and $\mathcal{B} = \langle S, \Sigma, s_0, \eta, G\rangle$. Recall that $q \sim s$ if there exists $w \in \Sigma^*$ such that $\delta(q_0, w) = q$ and $\eta(s_0, w) = s$. We say that $\mathcal{B}$ is an *abstraction* of $\mathcal{A}$ if for every $q \in Q$ there exists a single $s \in S$ such that $q \sim s$. An abstraction $\mathcal{B}$ of $\mathcal{A}$ is called a *miser abstraction of* $\mathcal{A}$ if $L(\mathcal{A}) \subseteq L(\mathcal{B})$, and the set G of $\mathcal{B}$'s accepting states cannot be reduced retaining the containment. That is, for all $s \in G$, the DFA $\mathcal{B}_s = \langle S, \Sigma, s_0, \eta, G \setminus \{s\}\rangle$ is such that $L(\mathcal{A}) \not\subseteq L(\mathcal{B}_s)$. It is not hard to see that $L(\mathcal{A}) \subseteq L(\mathcal{B})$ iff for every $q \in F$ and $s \in S$ such that $q \sim s$, it holds that $s \in G$. The above suggests a simple criterion for fixing the set of accepting states required for an abstraction to be miser.

Simple decompositions of a DFA consists of miser abstractions. Let $L \subseteq \Sigma^*$. Clearly, if L is simply-composite, then it is composite. The opposite is not necessarily true, as shown by the following example.

Example 6.1. Let $\mathcal{A}$ be the minimal DFA for the singleton language $\{\text{“}ab\text{”}\}$. According to Example 3.2, the DFA $\mathcal{A}$ is composite. However, one can go over all the miser abstractions of its 4-state DFA and verify that $\mathcal{A}$ is not simply-composite.

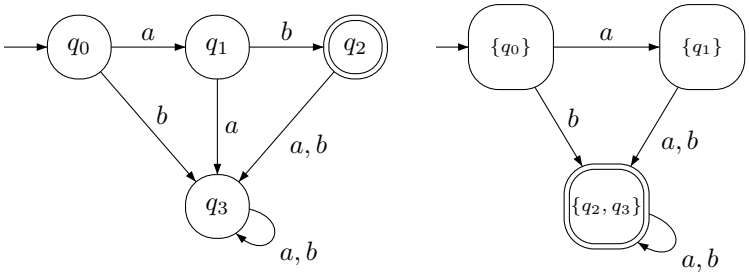

Figure 4: The language of the DFA $\mathcal{A}$ is $\{\text{“}ab\text{”}\}$. The DFA $\mathcal{B}$ is a miser abstraction of $\mathcal{A}$, obtained by merging the states q_2 and q_3. Had we changed the accepting state set of $\mathcal{B}$, it would have been an abstraction of $\mathcal{A}$, but not a miser abstraction.

Consider a DFA $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ and $t \in \mathbb{N}$. We use $\gamma_t(\mathcal{A})$ to denote the set of all miser abstractions of $\mathcal{A}$ with index at most t. Let $\mathit{ceiling}_t(\mathcal{A}) = \bigcap_{\mathcal{B}\in\gamma_t(\mathcal{A})} L(\mathcal{B})$.

[4] Note that a naive merge of states introduces nondeterminism. Here, merging states implies also a merge of their successors, thus determinism is retained.

Note that $\mathcal{A}$ is simply decomposable iff $L(\mathcal{A}) = ceiling_t(\mathcal{A})$ for $t = ind(\mathcal{A}) - 1$. Our next goal is an algorithm that decides whether a given DFA is simply-composite.

For $t \in \mathbb{N}$, a *t-partition* of Q is a set $P = \{Q_1, \ldots, Q_{t'}\}$ of $t' \leq t$ nonempty and pairwise disjoint subsets of Q whose union is Q. For $q \in Q$, we use $[q]_P$ to refer to the set Q_i such that $q \in Q_i$.

Definition 6.1. Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ be a DFA. A t-partition P of Q is a *good t-partition of $\mathcal{A}$* if δ respects P. That is, for every $q \in Q$, $\sigma \in \Sigma$, and $q' \in [q]_P$, we have that $\delta(q', \sigma) \in [\delta(q, \sigma)]_P$.

A good t-partition of $\mathcal{A}$ induces a miser abstraction of it. In the other direction, each abstraction of $\mathcal{A}$ with index at most t induces a t-partition of $\mathcal{A}$. Formally, we have the following.

Lemma 6.2. *There is a one-to-one correspondence between the DFAs in $\gamma_t(\mathcal{A})$ and the good t-partitions of $\mathcal{A}$.*

PROOF. Consider a DFA $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ such that every state of $\mathcal{A}$ is reachable. Let P be a good t-partition of $\mathcal{A}$. Define $\mathcal{B}_P = \langle P, \Sigma, [q_0]_P, \eta, G\rangle$, where η is naturally induced by δ and $G = \{Q_i \in P : Q_i \cap F \neq \emptyset\}$. Note that $\mathcal{B}_P$ is a miser abstraction of $\mathcal{A}$ and furthermore, that $\mathcal{B}_P \in \gamma_t(\mathcal{A})$. It is not hard to verify that the function from P to $\mathcal{B}_P$ is one to one and onto $\gamma_t(\mathcal{A})$.

Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ be a DFA and let $q \in Q \setminus F$. Let P be a good partition of $\mathcal{A}$. If $[q]_P \cap F = \emptyset$, we say that P is a *q-excluding partition*.

Lemma 6.3. *Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ be a DFA such that every state of $\mathcal{A}$ is reachable and $F \neq Q$. Then, the DFA $\mathcal{A}$ is t-simply-decomposable iff for every state $q \in Q \setminus F$ there exists a q-excluding t-partition.*

PROOF. Assume that for every $q \in Q \setminus F$, there exists a q-excluding t-partition P_q. Let $\mathcal{B}_q \in \gamma_t(\mathcal{A})$ be the DFA associated with the good partition P_q as in Lemma 6.2. Note that $ind(\mathcal{B}_q) = |P_q| \leq t$. It is straightforward to verify that $L(\mathcal{A}) = \bigcap_{q \in Q \setminus F} L(\mathcal{B}_q)$ and therefore, that $\mathcal{A}$ is t-simply-decomposable.

For the other direction, assume that $\mathcal{A}$ is t-simply-decomposable. Then, $L(\mathcal{A}) = \bigcap_{\mathcal{B} \in \gamma_t(\mathcal{A})} L(\mathcal{B})$. Let $q \in Q \setminus F$ and let $w \in \Sigma^*$ be such that $\delta(q_0, w) = q$. Note that such a word w exists since every state of$\mathcal{A}$ is reachable. Since $w \notin L(\mathcal{A})$, there exists a DFA $\mathcal{B} = \langle S, \Sigma, s_0, \eta, G\rangle \in \gamma_t(\mathcal{A})$ such that $w \notin L(\mathcal{B})$. Let P be the good partition of $\mathcal{A}$ defined by $\mathcal{B}$. Note that $|P| \leq ind(\mathcal{B}) \leq t$. Let $s \in S$ be the unique state of $\mathcal{B}$ satisfying $s \sim q$. Since $\gamma_t(\mathcal{A})$ contains only abstractions of $\mathcal{A}$, the state is indeed unique. Note that $\eta(s_0, w) = s$, thus $s \notin G$, which implies that $[q]_P \cap F = \emptyset$. Therefore, the partition P is a q-excluding partition, and we are done.

For two partitions P and P' of Q, we say that P' is a *refinement* of P if for every $R \in P$ there exist sets $R'_1 \ldots R'_s \in P'$ such that $R = \bigcup_{i=1}^{s} R'_i$.

Let $q, q' \in Q$ be such that $q \neq q'$. Let P be a good partition of $\mathcal{A}$. If $[q]_P = [q']_P$, we say that *P joins q and q'*.

It is not hard to see that good partitions are closed under intersection. That is, if P and P' are good partitions, so is the partition that contains the intersections of their members. Thus, there exists a unique good partition of $\mathcal{A}$, denoted $P_{q,q'}$ such that if P is a good partition of $\mathcal{A}$ that joins q and q', then $P_{q,q'}$ is a refinement of P. The following lemma states that the partition $P_{q,q'}$ can be computed efficiently.

Lemma 6.4. *Given a DFA $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ and $q, q' \in Q$ such that $q \neq q'$, the partition $P_{q,q'}$ can be computed in polynomial time.*

PROOF. We present a polynomial time algorithm to compute $P_{q,q'}$. The algorithm iteratively computes a sequence $P_0, P_1 \ldots P_m$ of partitions of $\mathcal{A}$. In the partition P_0, the states q and q' belong to the same set, and all other sets are singletons. In the i-th iteration, the algorithm searches for $s, s' \in Q$ and $\sigma \in \Sigma$ such that $[s]_{P_i} = [s']_{P_i}$ but $[\delta(s,\sigma)]_{P_i} \neq [\delta(s',\sigma)]_{P_i}$. If such s, s', and σ exist, the partition P_{i+1} is identical to the partition P_i, except for the sets $[\delta(s,\sigma)]_{P_i}$ and $[\delta(s',\sigma)]_{P_i}$, which are replaced by their union in P_{i+1}. If such s, s', and σ do not exist, the algorithm sets $m = i$ and returns P_m as $P_{q,q'}$.

First, note that the stopping condition of the algorithm implies that the partition P_m is a good partition of $\mathcal{A}$. Let P be a good partition of $\mathcal{A}$ such that $[q]_P = [q']_P$. Let $s, s' \in Q$ be such that $[s]_{P_m} = [s']_{P_m}$. We claim that $[s]_P = [s']_P$. This can be easily proven by induction on the minimal index i such that $[s]_{P_i} = [s']_{P_i}$. We have shown that P_m is a refinement of P. Therefore, $P = P_{q,q'}$.

To see that the algorithm runs in polynomial time, note that $m < |Q|$ and that each iteration runs in time polynomial in $|Q|$ and $|\Sigma|$.

Using Lemmas 6.3 and 6.4, we can now prove this section's main result.

Theorem 6.5. *Given a DFA $\mathcal{A}$, it is possible to decide in polynomial time whether $\mathcal{A}$ is simply-composite.*

PROOF. Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ and $k = ind(\mathcal{A})$. According to Lemma 6.3, the DFA $\mathcal{A}$ is simply-composite iff for every $s \in Q \setminus F$ there exists an s-excluding $(k-1)$-partition. Let P be a good $(k-1)$-partition. Since $\mathcal{A}$ has k states, there must exist $q, q' \in Q$ such that $q \neq q'$ and $[q]_P = [q']_P$. The partition $P_{q,q'}$ is thus a refinement of P. Therefore, if P is an s-excluding partition then $P_{q,q'}$ is an s-excluding partition. On the other hand, note that $|P_{q,q'}| \leq k-1$. Therefore, there exists an s-excluding $(k-1)$-partition iff there exists an s-excluding partition of the form $P_{q,q'}$ for some $q, q' \in Q$.

We now present a polynomial time algorithm to decide whether $\mathcal{A}$ is simply-composite.

1. For each $q, q' \in Q$, proceed as follows.
   (a) Compute $P_{q,q'}$.
   (b) For every $s \in Q \setminus F$, if $P_{q,q'}$ is an s-excluding partition, then mark s.
2. If all members of $Q \setminus F$ have been marked, then return true. Otherwise, return false.

The correctness of the algorithm is obvious from the above. Note that $P_{q,q'}$ can be computed in polynomial time based on Lemma 6.4, thus the algorithm runs in polynomial time.

Note that the above algorithm can be generalized to decide whether $\mathcal{A}$ is $(ind(\mathcal{A})-t)$-simply-decomposable in time exponential in t.

7. Algebraic Approach

In this section we use and develop concepts from the algebraic approach to automata in order to study the DFA primality problem. Utilizing this approach, we associate with every DFA $\mathcal{A}$, a monoid $M(\mathcal{A})$, and relate the properties of $M(\mathcal{A})$ with the primality of $\mathcal{A}$. We then focus on permutation DFAs – those for which the transition monoid is a group. By analyzing their transition monoid, we are able to show a simpler algorithm for checking the primality of permutation DFAs and to prove that composite permutation DFAs can be decomposed into permutation DFAs.

7.1. Definitions and notations

A *semigroup* is a set S together with an associative binary operation $\cdot : S \times S \to S$. A *monoid* is a semigroup S with an *identity element* $e \in S$ such that for every $x \in S$ it holds that $e \cdot x = x \cdot e = x$. Let S be a monoid with an identity element $e \in S$ and let $A \subseteq S$. Let A^* be the smallest monoid such that $A \subseteq A^*$ and $e \in A^*$. If $A^* = S$ we say that A *generates* S.

Consider a DFA $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$. For $w \in \Sigma^*$, let $\delta_w : Q \to Q$ be such that for every $q \in Q$, we have $\delta_w(q) = \delta(q, w)$. For $w_1, w_2 \in \Sigma^*$, the composition of δ_{w_1} and δ_{w_2} is, as expected, the operation $\delta_{w_2 \cdot w_1} : Q \to Q$ with $\delta_{w_2 \cdot w_1}(q) = \delta(q, w_2 \cdot w_1) = \delta_{w_1}(\delta_{w_2}(q))$. The set $\{\delta_w : w \in \Sigma^*\}$, equipped with the composition binary operation, is a monoid called the *transition monoid of* $\mathcal{A}$, denoted $M(\mathcal{A})$. Its identity element is δ_ϵ, denoted id. Note that $M(\mathcal{A})$ is generated by $\{\delta_\sigma : \sigma \in \Sigma\}$.

7.2. A monoid-driven characterization of primality

The following theorem and its corollary show that in order to decide whether a DFA is composite, we only need to know its state set, set of accepting states, and transition monoid. Thus, interestingly, changing the transition function or even the alphabet does not affect the composability of a DFA as long as the transition monoid remains the same.

Theorem 7.1. *Let $\mathcal{A}$ and $\mathcal{A}'$ be two DFAs with the same set of states, initial state, and set of accepting states. If $M(\mathcal{A}) = M(\mathcal{A}')$, then $depth(\mathcal{A}) = depth(\mathcal{A}')$.*

PROOF. We prove that $M(\mathcal{A}) \subseteq M(\mathcal{A}')$ implies that $depth(\mathcal{A}) \leq depth(\mathcal{A}')$. We then conclude that $M(\mathcal{A}) = M(\mathcal{A}')$ implies that $depth(\mathcal{A}) = depth(\mathcal{A}')$.

We parametrize the definition of $\alpha(\mathcal{A})$ by a bound t on the index of the DFAs in it. Thus, $\alpha_t(\mathcal{A}) = \{\mathcal{B} : \mathcal{B}$ is a minimal DFA with $ind(\mathcal{B}) \leq t$ and $L(\mathcal{A}) \subseteq L(\mathcal{B})\}$.

Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ and $\mathcal{A}' = \langle Q, \Sigma', q_0, \delta', F\rangle$. Consider the set $\delta_\Sigma = \{\delta_\sigma : \sigma \in \Sigma\}$ that generates $M(\mathcal{A})$. Since $M(\mathcal{A}) \subseteq M(\mathcal{A}')$, it holds that $\delta_\Sigma \subseteq M(\mathcal{A}')$. Therefore, there exists an encoding function $f : \Sigma \to \Sigma'^*$ such that for every $\sigma \in \Sigma$, we have $\delta_\sigma = \delta'_{f(\sigma)}$. We can extend the function f to the domain Σ^* as follows: For $w = w_1 \cdots w_m$ with $w_i \in \Sigma$, define $f(w) = \prod_{i=1}^{m} f(w_i)$. It is trivial to verify that for

every $w \in \Sigma^*$, it holds that $\delta_w = \delta'_{f(w)}$, so the run of $\mathcal{A}$ on w ends in the same state as the run of $\mathcal{A}'$ on $f(w)$. Since $\mathcal{A}$ and $\mathcal{A}'$ agree on the set of accepting states, it follows that $w \in L(\mathcal{A})$ iff $f(w) \in L(\mathcal{A}')$.

Now, let $t = depth(\mathcal{A}) + 1$, and let w be a t-primality witness for $L(\mathcal{A})$. That is, $w \in (\cap_{\mathcal{B} \in \alpha_t(\mathcal{A})} L(\mathcal{B})) \setminus L(\mathcal{A})$. We claim that $f(w)$ is a t-primality witness for $\mathcal{A}'$, thus proving the theorem. From the above, $w \notin L(\mathcal{A})$ implies $f(w) \notin L(\mathcal{A}')$. Consider a DFA $\mathcal{B}' \in \alpha_t(\mathcal{A}')$. Let $\mathcal{B}' = \langle S, \Sigma', s_0, \eta', G \rangle$. We define the function $\eta : S \times \Sigma \to S$ by $\eta(s, \sigma) = \eta'(s, f(\sigma))$. From the definition of the extension of f to Σ^*, it follows that for every $w \in \Sigma^*$, we have $\eta_w = \eta'_{f(w)}$. We now define the DFA $\mathcal{B} = \langle S, \Sigma, s_0, \eta, G \rangle$. Clearly, for every $w \in \Sigma^*$, it holds that $w \in L(\mathcal{B})$ iff $f(w) \in L(\mathcal{B}')$. Specifically, for every $w \in \Sigma^*$, we have that $w \in L(\mathcal{A})$ implies $f(w) \in L(\mathcal{A}')$. Then, as $\mathcal{B}' \in \alpha(\mathcal{A}')$ we have that $f(w) \in L(\mathcal{B}')$, implying that $w \in L(\mathcal{B})$. So, $L(\mathcal{A}) \subseteq L(\mathcal{B})$ and $\mathcal{B} \in \alpha_t(\mathcal{A})$. Since w is a t-primality witness for $L(\mathcal{A})$, we have that $w \in L(\mathcal{B})$, and therefore $f(w) \in L(\mathcal{B}')$, proving that $f(w)$ is indeed a t-primality witness for $\mathcal{A}'$.

Let Σ and Σ' be the alphabets of $\mathcal{A}$ and $\mathcal{A}'$, respectively. The fact $M(\mathcal{A}) \subseteq M(\mathcal{A}')$ enables us to "encode" every letter in Σ by a word in Σ' that acts the same way on the set of states. By expanding this encoding, we can encode every word over Σ by a word over Σ'. In particular, a primality witness for $\mathcal{A}$ is encoded into a primality witness for $\mathcal{A}'$.

Theorem 7.1 suggests that we can relate the properties of a DFAs transition monoid to the question of its primality. In the next section we do so for the family of *permutation DFAs*.

7.3. Permutation DFAs

Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F \rangle$ be a DFA. If for every $\sigma \in \Sigma$, it holds that δ_σ is a permutation, then $\mathcal{A}$ is called a *permutation DFA*. Equivalently, $\mathcal{A}$ is a permutation DFA if the monoid $M(\mathcal{A})$ is a group. It is easy to verify that the two definitions are indeed equivalent.

Note that a permutation DFA is strongly connected, unless it has unreachable states. From here on, we assume that all permutation DFAs we refer to are strongly connected.

Example 7.2. [The discrete cube DFA]: Let $n \in \mathbb{N}$. Consider the DFA $\mathcal{A}_n = \langle \mathbb{Z}_2^n, \mathbb{Z}_2^n, 0, \delta, \mathbb{Z}_2^n \setminus \{0\} \rangle$, where $\delta(x, y) = x + y$. Recall that $\mathbb{Z}_2^n = (0, 1)^n$. See, for example, the DFA $\mathcal{A}_2$ that appears in Figure 5.

The language of $\mathcal{A}_n$ is the set of all words $w_1 \dots w_m$ with $w_i \in \mathbb{Z}_2^n$ and $\sum_{i=1}^m w_i \neq 0$. It is easy to see that $\mathcal{A}_n$ is a permutation DFA of index 2^n and that it is minimal. We now prove that $\mathcal{A}_n$ is prime.

Consider a word $w = w_1 \dots w_m$ that satisfies the following two conditions:

- $w \notin L(\mathcal{A}_n)$. That is, $\sum_{i=1}^m w_i = 0$.
- The run of $\mathcal{A}_n$ on w visits all the states of $\mathcal{A}_n$.

Since $\mathcal{A}_n$ is a permutation DFA, such a word w exists. For example, for $n = 2$, one such possible word is

$$((1,0),(1,1),(1,0),(1,1)).$$

We claim that w is a primality witness for $\mathcal{A}_n$. Let $\mathcal{B} \in \alpha(\mathcal{A}_n)$ and set $\mathcal{B} = \langle Q, \mathbb{Z}_2^n, q_0, \eta, F\rangle$. Since $\mathcal{B}$ has less than 2^n states, and since the run of $\mathcal{A}$ on w visits all the states of $\mathcal{A}$, there exist $1 \leq i < j \leq m$ such that (i) $\delta(0, w_1 \dots w_i) \neq \delta(0, w_1 \dots w_j)$ but (ii) $\eta(q_0, w_1 \dots w_i) = \eta(q_0, w_1 \dots w_j)$. Let $z = w_1 \dots w_i \dots w_{j+1} \dots w_m$. From (ii), we have that $\eta(q_0, w) = \eta(q_0, z)$, thus $w \in L(\mathcal{B})$ iff $z \in L(\mathcal{B})$. From (i), we have

$$\delta(0,z) = \delta(0,w) - \delta(0, w_{i+1} \dots w_j) = \delta(0,w) + \delta(0, w_1 \dots w_i) - \delta(0, w_1 \dots w_j) \neq \delta(0,w) = 0.$$

Therefore, $z \in L(\mathcal{A}_n)$. Since $L(\mathcal{A}_n) \subseteq L(\mathcal{B})$, we have that $z \in L(\mathcal{B})$ and $w \in L(\mathcal{B})$, proving that w is indeed a primality witness for $\mathcal{A}_n$.

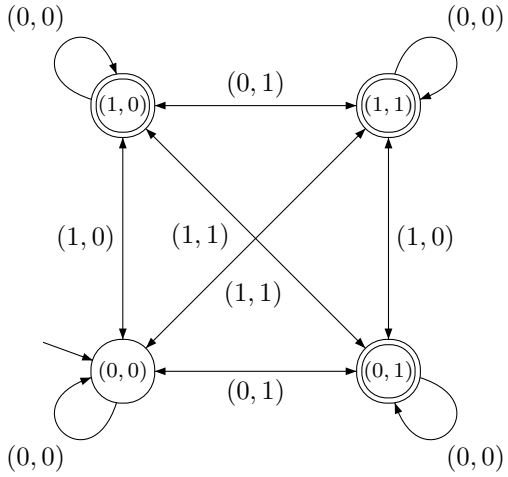

Figure 5: The discrete cube DFA $\mathcal{A}_2$.

We start working towards an analogue of Theorem 5.6 for permutation DFAs. Thus, our goal is to show that a composite permutation DFA can be decomposed using only permutation DFAs as factors. We first need some notations.

Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ be a DFA and let $f \in M(\mathcal{A})$. The *degree* of f, denoted $\deg(f)$, is $|f(Q)|$. The *degree* of $\mathcal{A}$ is $\deg(\mathcal{A}) = \min\{\deg(f) : f \in M(\mathcal{A})\}$.

The following are simple observations about degrees.

Proposition 7.3. *Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ be a DFA.*

1. *The DFA $\mathcal{A}$ is a permutation DFA iff* $\deg(\mathcal{A}) = |Q|$.
2. *Let $w, l, r \in \Sigma^*$. Then,* $\deg(\delta_w) \geq \deg(\delta_{l \cdot w \cdot r})$.
3. *Let $w \in \Sigma^*$. If* $\deg(\delta_w) = \deg(\mathcal{A})$*, then* $\deg(\delta_{r \cdot w \cdot l}) = \deg(\mathcal{A})$ *for all $r, l \in \Sigma^*$.*

We can now prove a variant of Theorem 5.6 with permutation DFAs replacing strongly connected ones.

Theorem 7.4. *Let $\mathcal{A}$ be a permutation minimal DFA and let $t \in \mathbb{N}$. If $\mathcal{A}$ is t-decomposable then it can be t-decomposed using only permutation DFAs as factors.*

PROOF. Let $\mathcal{A} = \langle Q, \Sigma, q_0\delta, F\rangle$. Since $\mathcal{A}$ is decomposable, there exist $\mathcal{A}_1 \dots \mathcal{A}_m$ with indices at most t, such that $L(\mathcal{A}) = \bigcap_{i=1}^{m} L(\mathcal{A}_i)$. Let $\mathcal{A}_i = \langle Q_i, \Sigma, q_0^i, \delta^i, F_i\rangle$. For each $1 \leq i \leq m$, let $w_i \in \Sigma^*$ be a word such that $\deg(\delta^i_{w_i}) = \deg(\mathcal{A}_i)$. Let $w' \in \Sigma^*$ be a word such that $\delta_{w_1 \cdot w_2 \cdots w_m \cdot w'}$ is the identity function on Q. Such a word w' exists since $M(\mathcal{A})$ is a group. Let $w = w_1 \cdot w_2 \cdots w_m \cdot w'$. Note that for every $1 \leq i \leq m$, it holds that $\deg(\delta^i_w) = \deg(\mathcal{A}_i)$.

Now, for every $1 \leq i \leq m$, define the DFA $\mathcal{B}_i = \langle S_i, \Sigma, \delta^i_w(q_0^i), \eta^i, F_i \cap S_i\rangle$, where $S_i = \delta^i_w(Q_i)$ and $\eta^i(q, z) = \delta^i(q, z \cdot w)$. Note that $\mathcal{B}_i$ is well defined, that it is a permutation DFA and that $ind(\mathcal{B}_i) \leq ind(\mathcal{A}_i)$. We claim that $L(\mathcal{A}) = \bigcap_{i=1}^{m} L(\mathcal{B}_i)$, proving that $\mathcal{A}$ can be $t-$decomposed using only permutation DFAs as factors.

Let $z \in \Sigma^*$ and assume that $z = z_1 \cdots z_t$ where $z_i \in \Sigma$. Consider the word $x = w \cdot z_1 \cdot w \cdot z_2 \cdot w \cdots w \cdot z_t \cdot w$. Note that for every $1 \leq i \leq m$, we have that $x \in L(\mathcal{A}_i)$ iff $z \in L(\mathcal{B}_i)$. On the other hand, note that $x \in L(\mathcal{A})$ iff $z \in L(\mathcal{A})$. Therefore, we have that $z \in L(\mathcal{A})$ iff $z \in \bigcap_{i=1}^{m} L(\mathcal{B}_i)$, and we are done.

Beyond its theoretical interest, Theorem 7.4 implies that when checking the primality of a permutation DFA, we may consider only permutation DFAs as candidate factors. We now use this result in order to develop a more efficient algorithm for deciding the primality of permutation DFAs. We first need some observations on permutation DFAs.

7.3.1. Observations on permutation DFAs

Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ be a permutation DFA. We say that $\mathcal{A}$ is *inverse-closed* if for every $\sigma \in \Sigma$ there exists a letter, denoted $\sigma^{-1} \in \Sigma$, such that $\delta_{\sigma^{-1}} = (\delta_\sigma)^{-1}$. When $\mathcal{A}$ is not inverse-closed, we can consider the *inverse-closure* of $\mathcal{A}$, which is the DFA $\mathcal{A}' = \langle Q, \Sigma', q_0, \delta', F\rangle$, where $\Sigma' = \Sigma \cup \{\sigma^{-1} : \sigma \in \Sigma\}$ and

$$\delta'(q, \tau) = \begin{cases} \delta(q, \tau) & \text{if } \tau \in \Sigma, \\ (\delta_\sigma)^{-1}(q) & \text{if } \tau = \sigma^{-1} \text{ for some } \sigma \in \Sigma. \end{cases}$$

Recall that $\mathcal{A}$ is a permutation DFA, and thus $(\delta_\sigma)^{-1}$ is well-defined.

Lemma 7.5. *Let $\mathcal{A}$ be a permutation DFA and let $\mathcal{A}'$ be the inverse-closure of $\mathcal{A}$. Then, $depth(\mathcal{A}) = depth(\mathcal{A}')$.*

PROOF. Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ and let $\mathcal{A}' = \langle Q, \Sigma', q_0, \delta', F\rangle$. Theorem 7.1 implies that it is enough to prove that $M(\mathcal{A}) = M(\mathcal{A}')$. Clearly, $M(\mathcal{A}) \subseteq M(\mathcal{A}')$. We need to show that $M(\mathcal{A}') \subseteq M(\mathcal{A})$. Let $\tau \in \Sigma'$. Note that it is enough to show that $\delta'_\tau \in M(\mathcal{A})$. If $\tau = \sigma$ for some $\sigma \in \Sigma$, then $\delta'_\tau = \delta_\sigma \in M(\mathcal{A})$. If $\tau = \sigma^{-1}$ for some $\sigma \in \Sigma$, note that $\delta_\sigma \in M(\mathcal{A})$. Since $M(\mathcal{A})$ is a group, we have that $\delta'_\tau = (\delta_\sigma)^{-1} \in M(\mathcal{A})$, and we are done.

From now on we assume that the permutation DFAs that we need to decompose are inverse-closed. By Lemma 7.5 we do not lose generality doing so.

Given an alphabet Σ, we use F_Σ to denote the group generated by $\Sigma \cup \{\sigma^{-1} : \sigma \in \Sigma\}$ with the only relations being $\sigma \cdot \sigma^{-1} = \sigma^{-1} \cdot \sigma = \epsilon$ for every $\sigma \in \Sigma$. We note that the group F_Σ is also known as the *free group* over Σ.

Given a permutation DFA $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$, we can think of F_Σ as a group acting on Q. Let $\tau \in \Sigma \cup \{\sigma^{-1} : \sigma \in \Sigma\}$. We describe the action of τ on Q by means of a function $\delta_\tau : Q \to Q$ defined as follows. If $\tau = \sigma \in \Sigma$, then $\delta_\tau(q) = \delta_\sigma(q)$. If $\tau = \sigma^{-1}$ with $\sigma \in \Sigma$, then $\delta_\tau(q) = \delta_\sigma^{-1}(q)$. Let $w \in F_\Sigma$ be such that $w = \tau_1 \cdots \tau_m$. The action of w on Q is then $\delta_w = \delta_{\tau_m} \cdots \delta_{\tau_1}$.

The *stabilizer subgroup* of q is the group $G(q) = \{w \in F_\Sigma : \delta_w(q) = q\}$. Let $w \in F_\Sigma$. The set $G(q) \cdot w$ is called a *right coset* of $G(q)$. The *index of* $G(q)$ *in* F_Σ, denoted $[F_\Sigma : G(q)]$, is defined as the number of right cosets of $G(q)$.

Proposition 7.6. *Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ be an inverse-closed permutation DFA of index n.*

1. *Let $w, z \in F_\Sigma$. Then, $\delta(q_0, w) = \delta(q_0, z)$ iff w and z belong to the same right coset of $G(q_0)$.*
2. *$[F_\Sigma : G(q_0)] = n$.*
3. *Let $q, q' \in Q$. Then, the subgroups $G(q)$ and $G(q')$ are conjugate. That is, there exists $w \in F_\Sigma$ such that $G(q) = w \cdot G(q') \cdot w^{-1}$. The opposite holds as well. That is, for every $q \in Q$ and $w \in \Sigma_F$ there exists some $q' \in Q$ such that $G(q) = w \cdot G(q') \cdot w^{-1}$.*

PROOF. Let $w, z \in F_\Sigma$. Note that w and z are in the same right coset of $G(q_0)$ iff $w \cdot z^{-1} \in G(q_0)$, which is true iff $\delta_{w \cdot z^{-1}}(q_0) = q_0$, which is equivalent to $\delta_w(q_0) = \delta_z(q_0)$. This proves the first claim in the proposition.

Let $q \in Q$. Since $\mathcal{A}$ is strongly connected, there exists a right coset of $G(q_0)$ of words $w \in F_\Sigma$ with $\delta_w(q_0) = q$. This proves the second claim in the proposition.

To prove the third claim, consider $q, q' \in Q$. Since $\mathcal{A}$ is strongly connected, there exists $w \in \Sigma^*$ such that $\delta(q, w) = q'$. We claim that $G(q) = w \cdot G(q') \cdot w^{-1}$. Since $G(q)$ and $w \cdot G(q') \cdot w^{-1}$ are both subgroups of index n, it is enough to show that $w \cdot G(q') \cdot w^{-1} \subseteq G(q)$. Let $z \in G(q')$. We claim that $w \cdot z \cdot w^{-1} \in G(q)$. Note that $\delta(q, w \cdot z \cdot w^{-1}) = \delta(q', z \cdot w^{-1}) = \delta(q', w^{-1}) = q$.

On the other hand, let $q \in Q$ and $w \in \Sigma_F$. Let $q' = \delta_w(q)$. We claim that $G(q) = w \cdot G(q') \cdot w^{-1}$. Again, it is enough to show that $w \cdot G(q') \cdot w^{-1} \subseteq G(q)$. Let $z \in G(q')$. We claim that $w \cdot z \cdot w^{-1} \in G(q)$. Indeed, $\delta(q, w \cdot z \cdot w^{-1}) = \delta(q', z \cdot w^{-1}) = \delta(q', w^{-1}) = q$.

Fix Σ and let G be a subgroup of F_Σ such that $[F_\Sigma : G] = n$. Let C be the set of right cosets of G and let $D \subseteq C$. The pair $\langle G, D\rangle$ induces the DFA $\mathcal{A} = \langle C, \Sigma, G, \delta, D\rangle$, where $\delta(\sigma, G \cdot w) = G \cdot w \cdot \sigma$. Note that $\mathcal{A}$ is well defined, that it is a permutation DFA, and that $ind(\mathcal{A}) = n$. Consider a word $w \in \Sigma^*$. Note that $w \in L(\mathcal{A})$ iff the coset $G \cdot w$ is a member of D. Finally, note that the stabilizer set of the initial state of $\mathcal{A}$ is equal to G. Accordingly, we have the following.

Lemma 7.7. *There is a one to one correspondence between permutation DFAs and pairs $\langle G, D\rangle$, where G is a subgroup of F_Σ of the DFA's index and D is a set of right cosets of G in F_Σ.*

Let G be a group and let H be a subgroup of G. The subgroup H is said to be a *normal subgroup* of G if for every $g \in G$ it holds that $g \cdot H \cdot g^{-1} = H$.

Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ be a permutation DFA. Consider the subgroup $G(q_0)$. If $G(q_0)$ is a normal subgroup of F_Σ, we say that $\mathcal{A}$ is a *normal DFA*. From Proposition 7.6, we have that $\mathcal{A}$ is normal iff for every $q, q' \in Q$, it holds that $G(q) = G(q')$. Equivalently, $\mathcal{A}$ is normal iff for every $w \in F_\Sigma$ such that δ_w has a fixed point, it holds that δ_w is the identity function on Q.

Normal DFAs form a small fragment even within the set of permutation DFAs. We introduce them now, so that we can later use them in our algorithm for primality checking of general permutation DFAs. Our first step is to better understand the decomposability of normal DFAs.

Lemma 7.8. *Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ be a normal permutation DFA and let $\mathcal{B} = \langle S, \Sigma, s_0, \eta, G\rangle$ be a permutation DFA. Let $q_1, q_2 \in Q$ and $s_1, s_2 \in S$ be such that $q_1 \sim s_1$, $q_1 \sim s_2$, and $q_2 \sim s_1$. Then, $q_2 \sim s_2$.*

PROOF. Let $\mathcal{C}$ be the product DFA of $\mathcal{A}$ and $\mathcal{B}$. Since $\mathcal{C}$ is a product of permutation DFAs, it is itself a permutation DFA. By the assumption that $q_1 \sim s_1$ and $q_1 \sim s_2$, there exists $w \in \Sigma^*$ such that the run of $\mathcal{C}$ on w starting in state (q_1, s_1) ends at (q_1, s_2). Therefore: $\delta(q_1, w) = q_1$ and $\eta(s_1, w) = s_2$. Since $\mathcal{A}$ is normal, we also have $\delta(q_2, w) = q_2$. Therefore, the run of $\mathcal{C}$ on w starting in state (q_2, s_1) ends at (q_2, s_2) and we are done.

The following example shows that Lemma 7.8 does not hold if we remove the assumption that $\mathcal{A}$ is normal.

Example 7.9. The DFAs $\mathcal{A}$ and $\mathcal{B}$ appearing in Figure 6 are connected permutation DFAs that are not normal and for which Lemma 7.8 does not hold.

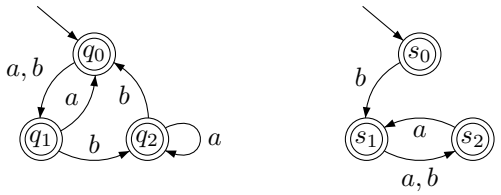

Figure 6: Lemma 7.8 does not hold for $\mathcal{A}$ and $\mathcal{B}$, which are not normal.

Lemma 7.10. *Let $\mathcal{A}$ be a normal permutation DFA and $\mathcal{B}$ be a permutation DFA such that $L(\mathcal{A}) \subseteq L(\mathcal{B})$. Then, there exists a permutation DFA $\mathcal{C}$ such that $L(\mathcal{A}) \subseteq L(\mathcal{C}) \subseteq L(\mathcal{B})$, $\mathcal{C}$ is an abstraction of $\mathcal{A}$, and $ind(\mathcal{C}) \leq ind(\mathcal{B})$.*

PROOF. Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F \rangle$ and $\mathcal{B} = \langle S, \Sigma, s_0, \eta, G \rangle$ be DFAs. For a pair of states $s, s' \in S$, we use $s \sim_{\mathcal{A}} s'$ in order to indicate that there exists $q \in Q$ such that $q \sim s$ and $q \sim s'$. From Lemma 7.8, it follows that $\sim_{\mathcal{A}}$ is an equivalence relation.

Now, define $\mathcal{C} = \langle S', \Sigma, s'_0, \eta', G' \rangle$ where:

- S' is the set of equivalence classes of $\sim_{\mathcal{A}}$.
- s'_0 is the equivalence class of s_0.
- η' is the transition function induced by η on S'. It is easy to verify that η' is well defined.
- $G' = \{s' \in S' : s' \subseteq G\}$.

It is now easy to verify that $L(\mathcal{C}) \subseteq L(\mathcal{B})$ and that $\mathcal{C}$ is an abstraction of $\mathcal{A}$. Clearly, $ind(\mathcal{C}) \leq ind(\mathcal{B})$, and so, we are done.

Using Lemma 7.10, we can now prove the following theorem about decompositions of normal DFAs.

Theorem 7.11. *Let $\mathcal{A}$ be a normal t-decomposable permutation DFA. Then $\mathcal{A}$ is t-simply-decomposable.*

PROOF. From Theorem 7.4, we have that $\mathcal{A}$ has a decomposition $L(\mathcal{A}) = \bigcap_{i=1}^{m} L(\mathcal{B}_i)$, such that for each $1 \leq i \leq m$, it holds that $\mathcal{B}_i$ is a permutation DFA and $ind(\mathcal{B}_i) \leq t$. From Lemma 7.10, there exists, for every $1 \leq i \leq m$, a permutation DFA $\mathcal{C}_i$ such that $L(\mathcal{A}) \subseteq L(\mathcal{C}_i) \subseteq L(\mathcal{B})$, the DFA $\mathcal{C}_i$ is an abstraction of $\mathcal{A}$, and $ind(\mathcal{C}_i) \leq ind(\mathcal{B}) \leq t$. Therefore, the decomposition $L(\mathcal{A}) = \bigcap_{i=1}^{m} L(\mathcal{C}_i)$ is a t-simple decomposition of $\mathcal{A}$.

Theorem 7.11 motivates us to replace a permutation DFA $\mathcal{A}$ by a normal permutation DFA. Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F \rangle$ be a DFA. We denote the *monoid DFA* of $\mathcal{A}$ by $\mathcal{A}_M = \langle M(\mathcal{A}), \Sigma, id, \delta_M, F_M \rangle$, with $\delta_M(f, w) = \delta_w \cdot f$ and $F_M = \{f \in M(\mathcal{A}) : f(q_0) \in F\}$. The following are simple observations about the monoid DFA. In particular, they show that the monoid DFA is normal, meeting our goal.

Proposition 7.12. *Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F \rangle$ be a DFA and let $\mathcal{A}_M$ be $\mathcal{A}$'s monoid DFA.*

1. $L(\mathcal{A}_M) = L(\mathcal{A})$.
2. $depth(\mathcal{A}_M) = depth(\mathcal{A})$.
3. *Every state in $\mathcal{A}_M$ is reachable.*
4. *Let $n = |Q|$. Then, $|M(\mathcal{A})| \leq n^n$.*
5. *If every state in $\mathcal{A}$ is reachable then $n \leq |M(\mathcal{A})|$.*
6. *If $\mathcal{A}$ is a permutation DFA, then $\mathcal{A}_M$ is a normal DFA.*

PROOF. Claims 1-5 are easy to verify. We will prove claim 6. Assume that $\mathcal{A}$ is a permutation DFA and let $\pi \in M(\mathcal{A})$. Let $w \in F_\Sigma$ such that $\delta_w(\pi) = \pi$. Recall from the definition of $\mathcal{A}_M$ that $\delta_w(\pi) = \delta_w \cdot \pi$. Therefore, we have $\delta_w \cdot \pi = \pi$. As $M(\mathcal{A})$ is a group, this implies that $\delta_w = id$. Thus, if δ_w has a fixed point, then it is the identity function, implying that $\mathcal{A}_M$ is normal.

Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ be a permutation DFA. For each $q \in Q$, let $G(q)$ be the stabilizer subgroup of q in $\mathcal{A}$. Let $\pi \in M(\mathcal{A})$ and let $G_M(\pi)$ be the stabilizer subgroup of π in $\mathcal{A}_M$. It holds that $G_M(\pi) = \bigcap_{q\in Q} G(q)$. For an arbitrary q, this group, the intersection of all members of a subgroup conjugacy class, is called the *normal core* of $G(q)$.

7.3.2. Deciding primality of permutation DFAs

The following theorem shows an immediate use of the transition monoid to prove the primality of a family of languages. Note that Example 7.2 is a special case of this more general theorem.

Theorem 7.13. *Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ be a permutation DFA of index n. If $|F| = n - 1$, then $\mathcal{A}$ is prime.*

PROOF. Assume by way of contradiction that $|F| = n - 1$ but $\mathcal{A}$ is composite. By Theorem 7.4, $\mathcal{A}$ can be decomposed using permutation DFAs. We show that all permutation DFAs in $\alpha(\mathcal{A})$ are trivial. That is, for every $\mathcal{B} \in \alpha(\mathcal{A})$ it holds that $L(\mathcal{B}) = \Sigma^*$. This leads to contradiction. Suppose that $\mathcal{B} \in \alpha(\mathcal{A})$ is a permutation DFA, and assume that $\mathcal{B} = \langle Q, \Sigma, q_0, \delta, F\rangle$ has at least 1 rejecting state. Then, the set of strings in F_Σ rejected by $\mathcal{B}$ contains a right coset $G(q_0)\cdot w$ of index strictly smaller than n. However, the set of rejected strings $comp(L(\mathcal{A}))$ in F_Σ is a coset of index n. A coset of index n cannot contain a coset of index smaller than n. Therefore, $comp(L(\mathcal{A}))$ cannot contain $G(q_0) \cdot w$, so $\mathcal{B} \notin \alpha(\mathcal{A})$, which is a contradiction.

We can now conclude with the following.

Theorem 7.14. *Deciding the primality of permutation DFA can be done in* PSPACE.

PROOF. Recall the following notation from the proof of Theorem 7.1.

$$\alpha_t(\mathcal{A}) = \{\mathcal{B} : \mathcal{B} \text{ is a minimal DFA with } ind(\mathcal{B}) \leq t \text{ and } L(\mathcal{A}) \subseteq L(\mathcal{B})\}.$$

Given $\mathcal{A}$, let $n = ind(\mathcal{A})$. The primality checking algorithm first checks whether $\mathcal{A}$ has unreachable states and removes them. Let $A_M = \langle M(\mathcal{A}), \Sigma, id, \delta_M, F_M\rangle$ be the monoid DFA of $\mathcal{A}$. From Proposition 7.12, we have that $depth(\mathcal{A}) = depth(\mathcal{A}_M)$. Hence, $\mathcal{A}$ is composite iff $depth(\mathcal{A}_M) \leq n - 1$ holds. From Theorem 7.11, we have that $depth(\mathcal{A}_M) \leq n-1$ iff $\mathcal{A}_M$ is $(n-1)$-simply-decomposable. Therefore, in order to decide whether $\mathcal{A}$ is composite, our algorithm only needs to decide whether $\mathcal{A}_M$ is $(n-1)$-simply-decomposable. This is done as follows:

1. For every $f \in M(\mathcal{A}) \setminus F_M$ and for each DFA $\mathcal{B} \in \alpha_{n-1}(\mathcal{A}_M)$:
   (a) Let $\mathcal{C}$ be the product DFA of $\mathcal{A}_M$ and $\mathcal{B}$. Check whether there is a reachable state in $\mathcal{C}$ of the form (f, s), where s is an accepting state of $\mathcal{B}$. If there is no such reachable state, mark f.
2. If every $f \in M(\mathcal{A})$ has been marked, accept. Otherwise, reject.

Assume that $\mathcal{A}_m$ is $(n-1)$-simply-decomposable. Then, according to Lemma 6.3, every $f \in M(\mathcal{A})\backslash F_M$ has an excluding $(n-1)$-partition P. Let $\mathcal{B} \in \gamma_{n-1}(\mathcal{A}_M)$ be the DFA corresponding to P. Note that on the iteration in which the algorithm considers $\mathcal{B}$, it marks the state f. Therefore, the algorithm accepts $\mathcal{A}_M$.

For the other direction, assume that the algorithm accepts $\mathcal{A}_M$. Then, for every $f \in M(\mathcal{A})\backslash F_M$ there exists $\mathcal{B}_f \in \alpha_{n-1}(\mathcal{A}_M)$ such that the state f was marked on the iteration in which the algorithm considered the DFA $\mathcal{B}_f$. We claim that $\bigcap_{f\in M(\mathcal{A})\backslash F_M} L(\mathcal{B}_f) = L(\mathcal{A}_M)$. Obviously, $\bigcap_{f\in M(\mathcal{A})\backslash F_M} L(\mathcal{B}_f) \subseteq L(\mathcal{A}_M)$. Now, let $w \in \bigcap_{f\in M(\mathcal{A})\backslash F_M} L(\mathcal{B}_f)$ and let $f \in M(\mathcal{A})$ be such that $\delta_M(id, w) = f$. Assume by way of contradiction that $f \notin F_M$. Let s be the state in which the run of $\mathcal{B}_f$ on w ends. Since $f \in L(\mathcal{B}_f)$, we have that s is an accepting state. Let $\mathcal{C}$ be the product DFA of $\mathcal{A}_M$ and $\mathcal{B}_f$. Note that the run of C on the word w ends in the state (f, s), where s is an accepting state of $\mathcal{B}_f$. This is in contradiction to our definition of $\mathcal{B}_f$. Therefore, it holds that $\bigcap_{f\in M(\mathcal{A})\backslash F_M} L(\mathcal{B}_f) = L(\mathcal{A}_M)$, and so, $\mathcal{A}_M$ is $(n-1)$-decomposable. As we have seen, this implies that $\mathcal{A}_M$ is $(n-1)$-simply decomposable, thus the algorithm is correct.

Since, by Lemma 7.12, it holds that $ind(\mathcal{A}_M) \leq n^n$, and the computation of $\mathcal{A}_M$ can proceed on-the-fly, the algorithm requires polynomial space.

8. Discussion

The motivation for this work has been compositional methods for LTL model checking and synthesis. Much to our surprise, we have realized that even the basic problem of DFA decomposition was still open. Not less surprising has been the big gap between the EXPSPACE and NLOGSPACE upper and lower bounds for the primality problem.

This article describes our struggle and partial success with the problem and this gap. While the general case is still open, we managed to develop some intuitions and tools that we believe to be interesting and useful, to develop helpful primality-related theory, to identify easy cases, and to develop an algebraic approach to the problem.

Our initial work focused on finding interesting examples for prime or composite families of regular languages. Some of these examples are presented in Sections 2 and 3.

In our attempt to develop a theory of DFA decomposition, we introduced two measures for the "decomposition complexity" of a composite regular language. These are the depth and the width. We also introduced the notion of a primality witness, whose minimal length is a complexity measure for the primality of a prime regular language. We conjecture that both the width and the minimal witness length have non-trivial upper bounds. The existence of such bounds could imply that there is a better upper bound than the one currently known for the primality problem.

While looking for an efficient algorithm for the general primality problem, we gathered interesting observations about certain fragments. Sometimes, these observations led to a more efficient algorithm for that fragment. Our results about simple co-safe DFAs where discovered while attempting to develop a primality checking algorithm based on the partition of the DFA into strongly connected components.

The notion of simple decomposition resulted from the natural idea of a decomposition algorithm that creates factors for a given DFA by merging states. It soon became clear that such an algorithm would be unable to express the full power of a DFA decomposition as we defined it. However, the notion of a simple decomposition turned out to be both interesting in its own right, and useful to us later when dealing with permutation DFAs.

The idea that we should consider primality of permutation DFAs as a fragment of the general DFA primality problem came after we studied the Krohn-Rhodes decomposition of DFAs [7]. Considering their decomposition, the immediately apparent difference is that Krohn and Rhodes seek to decompose a DFA into factors with a simpler algebraic structure (smaller transition monoid), while we seek to decompose a DFA into factors with a simpler combinatorial structure (less states). Searching for special cases where these two notions could be related, we started looking at permutation DFAs, as they have the benefit that their transition monoid is a group. Since the motivation for studying permutation DFAs came from their algebraic structure, it made sense to treat them with an algebraic approach. Indeed, this approach yielded several interesting results, including a PSPACE algorithm for primality checking of a permutation DFA.

8.1. Directions for Future Research

A challenge that has guided us through out this work was to determine the computational complexity of the DFA primality problem. There is still a wide gap between our lower and upper complexity bounds, and so, this problem remains open.

Our future work involves both further investigations of the theory and tools developed here and a study of richer settings of decomposition. In the first front, we seek results that bound (from both below and above) the length of a primality witness or bound the width of decompositions. Such results can give tighter bounds on the complexity of the primality problem.

Another approach is to try and bound the number of prime DFAs of length at most n. Since every composite DFA can be decomposed using only prime DFAs as factors, such a bound could lead to a bound on the width of composite DFAs. Furthermore such an insight about the density of prime DFAs within the set of all DFAs would be interesting in its own right.

In the second front, we study richer types of automata, mainly automata on infinite words (as with nondeterministic automata, an additional challenge in this setting is the lack of a canonical minimal automaton), as well as richer definitions of decomposition.

During this work, we have considered different notions of decomposability. One obvious decomposition is the *union decomposition*, in which we take the union of the factors instead of their intersection. Our results can be trivially dualized for union decomposition. Of greater interest is the *union-intersection decomposition*, in which one may apply not only intersection but also take the union of the underlying automata. This is a strongly stricter notion of decomposition than ours. Many of our results, for example primality checking of permutation DFAs, apply also in this stronger notion.

[1] B. Alpern and F.B. Schneider. Recognizing safety and liveness. *Distributed computing*, 2:117–126, 1987.

[2] W-P. de Roever, H. Langmaack, and A. Pnueli, editors. *Compositionality: The Significant Difference. Proceedings of Compositionality Workshop*, volume 1536 of *Lecture Notes in Computer Science*. Springer, 1998.

[3] P. Gazi. Parallel decompositions of finite automata. Master's thesis, Comenius University, Bratislava, Slovakia, 2006.

[4] Y.-S. Han, A. Salomaa, K. Salomaa, D. Wood, and S. Yu. On the existence of prime decompositions. *Theoretical Computer Science*, 376:60–69, 2007.

[5] J. Hartmanis and R.E Stearns. *Algebraic structure theory of sequential machines*. Prentice-Hall international series in applied mathematics. Prentice-Hall, 1966.

[6] N.D. Jones. Space-bounded reducibility among combinatorial problems. *Journal of Computer and Systems Science*, 11:68–75, 1975.

[7] K. Krohn and J. Rhodes. Algebraic theory of machines. i. prime decomposition theorem for finite semigroups and machines. *Transactions of the American Mathematical Society*, 116:450–464, 1965.

[8] O. Kupferman, N. Piterman, and M.Y. Vardi. Safraless compositional synthesis. In *Proc. 18th Int. Conf. on Computer Aided Verification*, volume 4144 of *Lecture Notes in Computer Science*, pages 31–44. Springer, 2006.

[9] J. Myhill. Finite automata and the representation of events. Technical Report WADD TR-57-624, pages 112–137, Wright Patterson AFB, Ohio, 1957.

[10] A. Nerode. Linear automaton transformations. *Proceedings of the American Mathematical Society*, 9(4):541–544, 1958.

[11] A. Pnueli. The temporal semantics of concurrent programs. *Theoretical Computer Science*, 13:45–60, 1981.

[12] A. Pnueli. Applications of temporal logic to the specification and verification of reactive systems: A survey of current trends. In *Proc. Advanced School on Current Trends in Concurrency*, pages 510–584. Volume 224, LNCS, Springer, 1985.

[13] M.Y. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1):1–37, 1994.