Prime Languages

Orna Kupferman and 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, methods to prove primality, and structural properties of DFAs that make the problem simpler or are retained in a decomposition.

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 [9]. 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 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 $\varphi_1, \ldots, \varphi_t$ [6]. 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 [11]. 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.

[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 [10].

In the algebraic approach, a DFA $\mathcal{A}$ is matched with a *monoid* $M(\mathcal{A})$. The members of $M(\mathcal{A})$ 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. 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. The algebraic approach is used in [5] 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 $\mathcal{A}$. 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 based on a syntactic congruence between words in Σ^*: given a regular language $L \subseteq \Sigma^*$, we have that $x \sim_L y$, for $x, y \in \Sigma^*$, 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 *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 [7,8], 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 smaller than the index of L. Otherwise, we say that L is *prime*. The definitions applies 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 proving 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. 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 the paper, 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, for example co-safety languages whose DFA contains one rejecting strongly connected component from which an accepting sink is reachable, where primality (with a short primality witness) is guaranteed. 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 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 [4] 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 primality. As [5], 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.

Due to lack of space, many examples and proofs are omitted. They can be found in the full version, in the authors' URLs.

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 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. [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 1. Let $\Sigma = \{a\}$ and $L_k = (a^k)^*$. 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, 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. $\square$

Let $\mathcal{A}$ be a DFA. We define $\alpha(\mathcal{A}) = \{\mathcal{B} : \mathcal{B} \text{ is a minimal DFA such that } \mathrm{L}(\mathcal{A}) \subseteq L(\mathcal{B}) \text{ and } ind(\mathcal{B}) < ind(\mathcal{A})\}$. That is, $\alpha(\mathcal{A})$ is the set of DFAs that contain $\mathcal{A}$ and have an index smaller than the index of $\mathcal{A}$. 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 1. *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 1. *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 $ind(roof(\mathcal{A})) \leq 2^{2^{O(n\cdot|\Sigma|\cdot\log n)}}$.*

Combining Theorem 1 and Lemma 1, we have the following.

Theorem 2. *The* PRIME-DFA *problem is in* EXPSPACE.

We note that the only lower bound for the problem is NLOGSPACE, by a reduction from reachability.

3 Primality Witnesses

Recall that a minimal DFA $\mathcal{A}$ is prime iff $roof(\mathcal{A}) \nsubseteq L(\mathcal{A})$. We define a *primality witness* for $\mathcal{A}$ as a word in $roof(\mathcal{A}) \setminus L(\mathcal{A})$. 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 L. Recall the language $L_k = (a^k)^*$ from Example 1. We show that the condition given there is necessary, thus 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 $k = p^r$. Since $w_k = a^{(p+1)p^{r-1}}$ is a primality witness for L_k, we can conclude that L_k is prime.

The bound on the size of $roof(\mathcal{A})$ from Lemma 1 implies the following.

Proposition 1. *A prime DFA has a primality witness of length doubly exponential.*

Proposition 1 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 1 to give a polynomial bound on the length of minimal primality witnesses, 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 2. 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 $w_n \notin L_n$. It can be shown that w_n is a primality witness for L_n. Hence, L_n is prime and a witness of length polynomial in $ind(L_n)$ exists. □

Example 3. Consider words $s = s_1 \cdots s_m$ and $w = w_1 \cdots w_t$, both over 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.

For, $w \in \Sigma^*$, let $L_w = \{s \in \Sigma^* : w \text{ is a subsequence of } s\}$. In the full version, we show that L_w is prime via a primality witness of length $2 \cdot ind(L_w)$. □

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. Let $\Sigma = \{a, b, c\}$ and let L be the language of prefixes of words in $c^*.(a^+.b^+.c^+)^*$. In the full version we show that L is composite with $width(L) = 3$. Also, it can be verified by a case-by-case analysis that L is not 2-factor composite. $\square$

Example 4 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.

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 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 $E = \{(q, q') : \exists \sigma \in \Sigma$ 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 2. *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, $L(\mathcal{A}^q) \subseteq L(\mathcal{B}^s)$.*

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 3. *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 $q_2 \sim s$.*

We are going to use Lemma 3 in our study of primality of classes of DFAs. We start with safe and co-safe DFAs.

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* [1] 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 simple co-safe.

Theorem 3. *Every simple co-safe DFA is prime with a primality witness of polynomial length.*

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

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. 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 $\mathcal{B} = \langle S, \Sigma, s_0, \eta, G \rangle$. 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. For example, while the singleton language $\{ab\}$ is composite, one can go over all the abstractions of its 4-state DFA and verify that $\mathcal{A}$ is not simply-composite. 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 $ceiling_t(\mathcal{A}) = \bigcap_{\mathcal{B} \in \gamma_t(\mathcal{A})} L(\mathcal{B})$. 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 3. *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 4. *There is a one-to-one correspondence between the DFAs in $\gamma_t(\mathcal{A})$ and the good t-partitions of $\mathcal{A}$.*

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 5. *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.*

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 partition $P_{q,q'}$ can be found by merging the two states q and q' and then merging only the pairs of states that must be merged in order for the generated partition to be good. Hence, the following holds.

Lemma 6. *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.*

By Lemma 5, we can decide whether $\mathcal{A}$ is simply-composite, by deciding, for every rejecting state q of $\mathcal{A}$, whether there exists a q-excluding $(ind(\mathcal{A}) - 1)$-partition. This can be checked by going over all partitions of the form $P_{s,s'}$ for some states s and s'. By Lemma 6, the latter can be done in polynomial time. We can thus conclude with the following.

Theorem 5. *Given a DFA $\mathcal{A}$, it is possible to decide in polynomial time whether $\mathcal{A}$ is simply-composite.*

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.

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 $\{\delta_\sigma : \sigma \in \Sigma\}$.

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 6. *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$(\mathcal{A}')$.*

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 6 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*.

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 5. [The discrete cube DFA]: Let $n \in \mathbb{N}$. Recall that $\mathbb{Z}_2^n = (0,1)^n$. 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 $\delta(x, y) = x + y$. The language of $\mathcal{A}_n$ is the set of all words $w_1 \ldots w_m$ 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. In the full version we show that $\mathcal{A}_n$ is prime. $\square$

We start working towards an analogue of Theorem 4 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 $\deg(\mathcal{A}) = \min\{\deg(f) : f \in M(\mathcal{A})\}$. Let $\mathcal{A}_i$ be a factor in a decomposition of $\mathcal{A}$ and let $w \in \Sigma^*$ be a word such that $\deg(\delta_w) = \deg(\mathcal{A}_i)$. Using simple observations about degrees, we can define a DFA $\mathcal{B}_i$ that is similar to $\mathcal{A}_i$ except that each letter σ acts in $\mathcal{B}_i$ as the word $\sigma \cdot w$ acts in $\mathcal{A}_i$. It can then be shown that $\mathcal{B}_i$ is a permutation DFA, and that it can replace $\mathcal{A}_i$ in the decomposition of $\mathcal{A}$. Hence the following theorem.

Theorem 7. *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.*

Beyond its theoretical interest, Theorem 7 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.

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 $\delta'(q,\tau)$ is $\delta(q,\tau)$ if $\tau \in \Sigma$, and is $(\delta_\sigma)^{-1}(q)$ if $\tau = \sigma^{-1}$ for some $\sigma \in \Sigma$.

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

Lemma 7. *Let $\mathcal{A}$ be a permutation DFA and let $\mathcal{A}'$ be the inverse-closure of $\mathcal{A}$. Then, $depth(\mathcal{A}) = depth(\mathcal{A}')$.*

From now on we assume that the permutation DFAs that we need to decompose are inverse-closed. By Lemma 7 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 $G(q) = \{w \in F_\Sigma : \delta_w(q_0) = q_0\}$. 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 $G(q)$.

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 8. *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*. It can be shown 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.

Lemma 9. *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, $q_2 \sim s_2$.*

Lemma 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 $ind(\mathcal{C}) \leq ind(\mathcal{B})$.*

Theorem 8. *Let $\mathcal{A}$ be a normal t-decomposable permutation DFA. Then $\mathcal{A}$ is t-simply-decomposable.*

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 $F_M = \{f \in M(\mathcal{A}) : f(q_0) \in F\}$. Simple observations about the monoid DFA show that $\mathcal{A}_M$ is normal and has the same depth as $\mathcal{A}$, meeting our goal

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 $G_M(\pi) = \bigcap_{q \in Q} G(q)$.

The following theorem shows an immediate use of the transition monoid to prove the primality of a family of languages. Note that Example 5 is a special case of this more general theorem.

Theorem 9. *Let $\mathcal{A} = \langle Q, \Sigma, q_0, \delta, F\rangle$ be a permutation DFA. If $|F| = ind(\mathcal{A}) - 1$, then $\mathcal{A}$ is prime.*

Consider a permutation DFA $\mathcal{A}$ of index n. Let $\mathcal{A}_M$ be its monoid DFA. By the above, instead of checking whether $\mathcal{A}$ is prime, we can check whether $\mathcal{A}_M$ is $(n-1)$-decomposable. Since $\mathcal{A}_M$ is normal, this is equivalent to asking whether $\mathcal{A}_M$ is $(n-1)$-simply-decomposable. As $\mathcal{A}_M$ is of size at most exponential in n, this check can be done in PSPACE. We thus have the following.

Theorem 10. *Deciding the primality of permutation DFA can be done in* PSPACE.

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 work described 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 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. 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. In particular, we are interested in union-intersection decompositions, where one may apply not only intersection but also take the union of the underlying automata. While it is easy to dualize our results for the case of union-only decomposition, mixing union and intersection results is a strictly stronger notion. Some of our results, for example the decomposition of permutation DFAs, apply also in this stronger notion.

References

1. Alpern, B., Schneider, F.B.: Recognizing safety and liveness. Distributed Computing 2, 117–126 (1987)
2. de Roever, W.-P., Langmaack, H., Pnueli, A. (eds.): COMPOS 1997. LNCS, vol. 1536. Springer, Heidelberg (1998)
3. Gazi, P.: Parallel decompositions of finite automata. Master's thesis, Comenius University, Bratislava, Slovakia (2006)
4. Hartmanis, J., Stearns, R.E.: Algebraic structure theory of sequential machines. Prentice-Hall international series in applied mathematics. Prentice-Hall (1966)
5. Krohn, K., Rhodes, J.: Algebraic theory of machines. i. prime decomposition theorem for finite semigroups and machines. Transactions of the American Mathematical Society 116, 450–464 (1965)
6. Kupferman, O., Piterman, N., Vardi, M.Y.: Safraless compositional synthesis. In: Ball, T., Jones, R.B. (eds.) CAV 2006. LNCS, vol. 4144, pp. 31–44. Springer, Heidelberg (2006)
7. Myhill, J.: Finite automata and the representation of events. Technical Report WADD TR-57-624, pp. 112–137, Wright Patterson AFB, Ohio (1957)
8. Nerode, A.: Linear automaton transformations. Proceedings of the American Mathematical Society 9(4), 541–544 (1958)
9. Pnueli, A.: The temporal semantics of concurrent programs. TCS 13, 45–60 (1981)
10. Pnueli, A.: Applications of temporal logic to the specification and verification of reactive systems: A survey of current trends. In: Rozenberg, G., de Bakker, J.W., de Roever, W.-P. (eds.) Current Trends in Concurrency. LNCS, vol. 224, pp. 510–584. Springer, Heidelberg (1986)
11. Vardi, M.Y., Wolper, P.: Reasoning about infinite computations. I&C 115(1), 1–37 (1994)