Information and Computation 207 (2009)1096–1118

Contents lists available at ScienceDirect

Information and Computation

journal homepage: www.elsevier.com/locate/ic

Detecting palindromes, patterns and borders in regular languages

Terry Anderson[a], John Loftus[b], Narad Rampersad[a,*], Nicolae Santean[a,1], Jeffrey Shallit[a]

[a] School of Computer Science, University of Waterloo, Waterloo, Ont., Canada N2L 3G1
[b] Luzerne County Community College, 1333 South Prospect Street, Nanticoke, PA 18634, USA

ARTICLE INFO

Article history:
Received 5 June 2008
Available online 19 March 2009

Keywords:
Palindrome
k-Power
Primitive word
Pattern
Bordered word

ABSTRACT

Given a language L and a non-deterministic finite automaton M, we consider whether we can determine efficiently (in the size of M) if M accepts at least one word in L, or infinitely many words. Given that M accepts at least one word in L, we consider how long a shortest word can be. The languages L that we examine include the palindromes, the non-palindromes, the k-powers, the non-k-powers, the powers, the non-powers (also called primitive words), the words matching a general pattern, the bordered words, and the unbordered words.

1. Introduction

Let $L \subseteq \Sigma^*$ be a fixed language, and let M be a deterministic finite automaton (DFA) or non-deterministic finite automaton (NFA) with input alphabet Σ. In this paper, we are interested in three questions:

1. Can we efficiently decide (in terms of the size of M) if $L(M)$ contains at least one element of L, that is, if $L(M) \cap L \neq \emptyset$?
2. Can we efficiently decide if $L(M)$ contains infinitely many elements of L, that is, if $L(M) \cap L$ is infinite?
3. Given that $L(M)$ contains at least one element of L, what is a good upper bound on a shortest element of $L(M) \cap L$?

We can also ask the same questions about $\overline{L}$, the complement of L.

As an example, consider the case where $\Sigma = \{\mathtt{a}\}$, L is the set of primes written in unary, that is, $\{\mathtt{a}^i : i \text{ is prime}\}$, and M is a NFA with n states.

To answer questions (1) and (2), we first rewrite M in Chrobak normal form [5]. Chrobak normal form consists of an NFA M' with a "tail" of $O(n^2)$ states, followed by a single non-deterministic choice to a set of disjoint cycles containing at most n states. Computing this normal form can be achieved in $O(n^5)$ steps by a result of Martinez [23].

Now we examine each of the cycles produced by this transformation. Each cycle accepts a finite union of sets of the form $(\mathtt{a}^t)^*\mathtt{a}^c$, where t is the size of the cycle and $c \leqslant n^2 + n$; both t and c are given explicitly from M'. Now, by Dirichlet's theorem on primes in arithmetic progressions, $\gcd(t, c) = 1$ for at least one pair (t, c) induced by M' if and only if M accepts infinitely many elements of L. This can be checked in $O(n^2)$ steps, and so we get a solution to question (2) in polynomial time.

Question (1) requires a little more work. From our answer to question (2), we may assume that $\gcd(t, c) > 1$ for all pairs (t, c), for otherwise M accepts infinitely many elements of L and hence at least one element. Each element in such a set

[*] Corresponding author. Present address: Department of Mathematics and Statistics, University of Winnipeg, 515 Portage Ave., Winnipeg, Man., Canada R3B 2E9.

E-mail addresses: tanderson@uwaterloo.ca (T. Anderson), jloftus@luzerne.edu (J. Loftus), n.rampersad@uwinnipeg.ca (N. Rampersad), nsantean@iusb.edu (N. Santean), shallit@graceland.uwaterloo.ca (J. Shallit).

[1] Department of Computer and Information Sciences, Indiana University South Bend, 1700 Mishawaka Ave., P.O. Box 7111, South Bend, IN 46634, USA.

0890-5401/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.ic.2008.06.007

is of length $kt + c$ for some $k \geqslant 0$. Let $d = \gcd(t, c) \geqslant 2$. Then $kt + c = (kt/d + c/d)d$. If $k > 1$, this quantity is at least $2d$ and hence composite. Thus it suffices to check the primality of c and $t + c$, both of which are at most $n^2 + 2n$. We can precompute the primes $<n^2 + 2n$ in $O(n^2)$ time using a modification of the sieve of Eratosthenes [25], and check if any of them are accepted. This gives a solution to question (1) in polynomial time.

On the other hand, answering question (3) essentially amounts to estimating the size of the least prime in an arithmetic progression, an extremely difficult question that is still not fully resolved [13], although it is known that there is a polynomial upper bound.

Even the case where L is regular can be difficult. Suppose L is represented as the complement of a language accepted by an NFA M' with n states. Then if $L(M) = \Sigma^*$, question (1) amounts to asking if $L(M') \neq \Sigma^*$, which is PSPACE-complete [2, Section 10.6]. Question (2) amounts to asking if $\overline{L(M')}$ is infinite, which is also PSPACE-complete [18]. Question (3) amounts to asking for good bounds on the smallest string not accepted by an NFA. There is an evident upper bound of 2^n, and there are examples known that achieve 2^{cn} for some constant $c > 0$, but more detailed analysis is still lacking [9].

Thus we see that asking these questions, even for relatively simple languages L, can quickly take us to the limits of what is known in formal language theory and number theory.

In this paper, we examine questions (1)–(3) in the case where M is an NFA and L is either the set of palindromes, the set of k-powers, the set of powers, the set of words matching a general pattern, the set of bordered words, or their complements.

In some of these cases, there is previous work. For example, Ito et al. [17] studied several circumstances in which primitive words (non-powers) may appear in regular languages. As a typical result in [17], we mention: "A DFA over an alphabet of two or more letters accepts a primitive word iff it accepts one of length $\leqslant 3n - 3$, where n is the number of states of the DFA". Horváth et al. [15] addressed the decidability problem of whether a language accepted by an NFA is palindromic (i.e., every element is a palindrome). They showed that the language accepted by an NFA with n states is palindromic if and only if all its words of length shorter than $3n$ are palindromes.

Here is a summary of the rest of the paper. In Section 2, we define the objects of study and our notation.

In Section 3, we begin our study of palindromes. We give efficient algorithms to test if an NFA accepts at least one palindrome, or infinitely many. We also show that a shortest palindrome accepted is of length at most quadratic, and further, that quadratic examples exist. In Section 4, we give efficient algorithms to test if an NFA accepts at least one non-palindrome, or infinitely many. Further, we give a tight bound on the length of a shortest non-palindrome accepted.

In Section 5, we begin our study of patterns. We show that it is PSPACE-complete to test if a given NFA accepts a word matching a given pattern. As a special case of this problem we consider testing if an NFA accepts a k-power. We give a algorithm to test if a k-power is accepted that is polynomial in k. If k is not fixed, the problem is PSPACE-complete. We also study the problem of accepting a power of exponent $\geqslant k$, and of accepting infinitely many k-powers.

In Section 6, we give a polynomial-time algorithm to decide if a non-k-power is accepted. We also give upper and lower bounds on the length of a shortest k-power accepted. In Section 7, we give an efficient algorithm for determining if an NFA accepts at least one non-power. In Section 8, we bound the length of the smallest power. Section 9 gives some additional results on powers.

In Section 10, we show how to test if an NFA accepts a bordered word, or infinitely many, and show that a shortest bordered word accepted can be of quadratic length. In Section 11 we give an algorithm to test if an NFA accepts an unbordered word, or infinitely many, and we establish a linear upper bound on the length of a shortest unbordered word.

2. Notions and notation

Let Σ be an alphabet, i.e., a non-empty, finite set of symbols (letters). By Σ^* we denote the set of all finite words (strings of symbols) over Σ, and by ϵ, the empty word (the word having zero symbols). The operation of concatenation (juxtaposition) of two words u and v is denoted by $u \cdot v$, or simply uv. If $w \in \Sigma^*$ is written in the form $w = xy$ for some $x, y \in \Sigma^*$, then the word yx is said to be a *conjugate* of w.

For $w \in \Sigma^*$, we denote by w^R the word obtained by reversing the order of symbols in w. A *palindrome* is a word w such that $w = w^R$. If L is a language over Σ, i.e., $L \subseteq \Sigma^*$, we say that L is *palindromic* if every word $w \in L$ is a palindrome.

Let $k \geqslant 2$ be an integer. A word y is a *k-power* if y can be written as $y = x^k$ for some non-empty word x. If y cannot be so written for any $k \geqslant 2$, then y is *primitive*. A 2-power is typically referred to as a *square*, and a 3-power as a *cube*.

Patterns are a generalization of powers. A *pattern* is a non-empty word p over a *pattern alphabet* Δ. The letters of Δ are called *variables*. A pattern p *matches* a word $w \in \Sigma^*$ if there exists a non-erasing morphism $h : \Delta^* \to \Sigma^*$ such that $h(p) = w$. Thus, a word w is a k-power if it matches the pattern a^k.

Bordered words are generalizations of powers. We say a word x is *bordered* if there exist words $u \in \Sigma^+$, $w \in \Sigma^*$ such that $x = uwu$. In this case, the word u is said to be a *border* for x. Otherwise, x is *unbordered*.

A *non-deterministic finite automaton* (NFA) over Σ is a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$ where Q is a finite set of states, $\delta : Q \times \Sigma \to 2^Q$ is a next-state function, q_0 is an initial state and $F \subseteq Q$ is a set of final states. We sometimes view δ as a transition table, i.e., as a set consisting of tuples (p, a, q) with $p, q \in Q$ and $a \in \Sigma$. The machine M is *deterministic* (DFA) if δ is a function mapping $Q \times \Sigma \to Q$. We consider only *complete* DFAs, that is, those whose transition function is a total function. Sometimes we use NFA-ϵ, which are NFAs that also allow transitions on the empty word.

The *size* of M is the total number N of its states and transitions. When we want to emphasize the components of M, we say M has n states and t transitions, and define $N := n + t$. The language of M, denoted by $L(M)$, belongs to the family of *regular languages* and consists of those words accepted by M in the usual sense. A *successful path*, or *successful computation* of M is any computation starting in the initial state and ending in a final state. The label of a computation is the input word that triggered it; thus, the language of M is the set of labels of all successful computations of M.

A state of M is *accessible* if there exists a path in the associated transition graph, starting from q_0 and ending in that state. By convention, there exists a path from each state to itself labeled with ϵ. A state q is *coaccessible* if there exists a path from q to some final state. A state which is both accessible and coaccessible is called *useful*, and if it is not coaccessible it is called *dead*.

We note that if M is an NFA or NFA-ϵ, we can remove all states that are not useful in linear time (in the number of states and transitions) using depth-first search. We observe that $L(M) \neq \emptyset$ if and only if any states remain after this process, which can be tested in linear time. Similarly, if M is a NFA, then $L(M)$ is infinite if and only if the corresponding digraph has a directed cycle. This can also be tested in linear time.

If M is an NFA-ϵ, then to check if $L(M)$ is infinite we need to know not only that the corresponding digraph has a cycle, but that it has a cycle labeled by a non-empty word. This can also be checked in linear time as follows. Let us suppose that all non-useful states of M have been removed. We wish to test whether there is some edge of the digraph of M that is part of some cycle and is not labeled by the empty word. We now observe that an edge of a digraph belongs to a directed cycle if and only if both of its endpoints lie within the same strongly connected component. It is well known that the strongly connected components of a graph can be computed in linear time (see [6, Section 22.5]). Once the strongly connected components of the NFA-ϵ are known, we simply check the edges not labeled by ϵ to determine if there is such an edge with both endpoints in the same strongly connected component. Thus we can determine if $L(M)$ is infinite in linear time.

Although the results of this paper are generally stated as applying to NFA's, by virtue of the preceding algorithm, one sees that the results apply equally well to NFA-ϵ's.

We will also need the following well-known results [14]:

Theorem 1. *Let M be an NFA with n states. Then*

(a) *$L(M) \neq \emptyset$ if and only if M accepts a word of length $< n$.*
(b) *$L(M)$ is infinite if and only if M accepts a word of length ℓ, $n \leqslant \ell < 2n$.*

If $L \subseteq \Sigma^*$ is a language, the *Myhill–Nerode equivalence relation* $\equiv_L$ is the equivalence relation defined as follows: for $x, y \in \Sigma^*$, $x \equiv_L y$ if for all $z \in \Sigma^*$, $xz \in L$ if and only if $yz \in L$. The classical Myhill–Nerode theorem asserts that if L is regular, the equivalence relation $\equiv_L$ has only finitely many equivalence classes.

For a background on finite automata and regular languages we refer the reader to Yu [33].

3. Testing if an NFA accepts at least one palindrome

Over a unary alphabet, every string is a palindrome, so problems (1)–(3) become trivial. Let us assume, then, that the alphabet Σ contains at least two letters. Although the palindromes over such an alphabet are not regular, the language

$$\left\{x \in \Sigma^* \ : \ xx^R \in L(M) \text{ or there exists } a \in \Sigma \text{ such that } xax^R \in L(M)\right\}$$

is, in fact, regular, as is often shown in a beginning course in formal languages [14, p. 72, Exercise 3.4 (h)]. We can take advantage of this as follows:

Lemma 2. *Let M be an NFA with n states and t transitions. Then there exists an NFA-ϵ M' with $n^2 + 1$ states and $\leqslant 2t^2$ transitions such that*

$$L(M') = \left\{x \in \Sigma^* \ : \ xx^R \in L(M) \textit{ or there exists } a \in \Sigma \textit{ such that } xax^R \in L(M)\right\}.$$

Proof. Let $M = (Q, \Sigma, \delta, q_0, F)$ be an NFA with n states. We construct an NFA-ϵ $M' = (Q', \Sigma, \delta', q_0', F')$ as follows: we let $Q' = Q \times Q \cup \{q_0'\}$, where q_0' is the new initial state, and we define the set of final states by

$$F' = \{[p,p] \ : \ p \in Q\} \cup \{[p,q] \ : \ \text{there exists } a \in \Sigma \text{ such that } q \in \delta(p,a)\}.$$

The transition function δ' is defined as follows:

$$\delta'(q_0', \epsilon) = \{[q_0, q] \ : \ q \in F\}$$

and

$$\delta'([p,q], a) = \{[r,s] \ : \ r \in \delta(p,a) \text{ and } q \in \delta(s,a)\}.$$

It is clear that M' accepts the desired language and consists of at most $n^2 + 1$ states and $2t^2$ transitions. □

Corollary 3. *Given an NFA M with n states and t transitions, we can determine if M accepts a palindrome in $O(n^2 + t^2)$ time.*

Proof. We create M' as in the proof of Lemma 2, and remove all states that are not useful, and their associated transitions. Now M accepts at least one palindrome if and only if $L(M') \neq \emptyset$, which can be tested in time linear in the number of transitions and states of M'. □

From Lemma 2, we obtain two other interesting corollaries.

Corollary 4. *Given an NFA M, we can determine if $L(M)$ contains infinitely many palindromes in quadratic time.*

Proof. We create M' as in the proof of Lemma 2, and remove all states that are not useful, and their associated transitions. M accepts infinitely many palindromes if and only if $L(M')$ is infinite, which can be tested in linear time, as described in Section 2. □

Corollary 5. *If an NFA M accepts at least one palindrome, it accepts a palindrome of length $\leqslant 2n^2 - 1$.*

Proof. Suppose M accepts at least one palindrome. Then M', as defined in Lemma 2, accepts at least one word. Although M' has $n^2 + 1$ states, the only transition from the initial state q_0' is an ϵ-transition to one of the other n^2 states. Thus if M' accepts a word, it must accept a word of length $\leqslant n^2 - 1$. Then M accepts either ww^R or waw^R, and both are palindromes, so M accepts a palindrome of length at most $2(n^2 - 1) + 1 = 2n^2 - 1$. □

For a different proof of this corollary, see Rosaz [28].

We observe that the quadratic bound is tight, up to a multiplicative constant, in the case of alphabets with at least two letters, and even for DFAs.

Proposition 6. *For infinitely many n there exists a DFA M with n states over a 2-letter alphabet such that*

(a) *M has n states.*
(b) *The shortest palindrome accepted by M_n is of length $\geqslant n^2/2 - 3n + 5$.*

Proof. For $t \geqslant 2$, consider the language $L_t = (\mathtt{a}^t)^+\mathtt{b}(\mathtt{a}^{t-1})^+$. This language evidently can be accepted by a DFA with $n = 2t + 2$ states. For a word $w \in L_t$ to be a palindrome, we must have $w = \mathtt{a}^{c_1 t}\mathtt{b}\mathtt{a}^{c_2(t-1)}$, for some integers $c_1, c_2 \geqslant 1$, with $c_1 t = c_2(t-1)$. Since t and $t-1$ are relatively prime, we must have $t - 1 | c_1$ and $t | c_2$. Thus the shortest palindrome in L_n is $\mathtt{a}^{t(t-1)}\mathtt{b}\mathtt{a}^{t(t-1)}$, which is of length $2t^2 - 2t + 1 = n^2/2 - 3n + 5$. □

4. Testing if an NFA accepts at least one non-palindrome

In this section, we consider the problem of deciding if an NFA accepts at least one non-palindrome. Evidently, if an NFA fails to accept a non-palindrome, it must accept nothing but palindromes, and so we discuss the opposite decision problem,

Given an NFA M, is $L(M)$ palindromic?

Again, the problem is trivial for a unary alphabet, so we assume $|\Sigma| \geqslant 2$.

Horváth et al. [15] proved that the question is recursively solvable. In particular, they proved the following theorem.

Theorem 7. *$L(M)$ is palindromic if and only if $\{x \in L(M) \ : \ |x| < 3n\}$ is palindromic, where n is the number of states of M.*

While a naive implementation of Theorem 7 would take exponential time, in this section we show how to test palindromicity in polynomial time. We also show the bound of $3n$ in Theorem 7 is tight for NFAs, and we improve the bound for DFAs.

First, we show how to construct a "small" NFA M_s', for some integer $s > 1$, that has the following properties:

(a) no word in $L(M_s')$ is a palindrome;
(b) M_s' accepts all non-palindromes of length $< s$ (in addition to some other non-palindromes).

The idea in this construction is the following: on input w of length $r < s$, we "guess" an index i, $1 \leqslant i \leqslant r/2$, such that $w[i] \neq w[r+1-i]$. We then "verify" that there is indeed a mismatch i characters from each end. We can re-use states, as illustrated in Fig. 1 for the case $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ and $s = 10$.

The resulting NFA M_s' has $O(|\Sigma|s)$ states and $O(|\Sigma|^2 s)$ transitions. A similar construction appears in [31].

Given an NFA M with n states, we now construct the cross-product with M_{3n}', and obtain an NFA A that accepts $L(M) \cap L(M_{3n}')$. We claim that $L(A) = \emptyset$ if and only if $L(M)$ is palindromic. For if $L(A) = \emptyset$, then M accepts no non-palindrome of

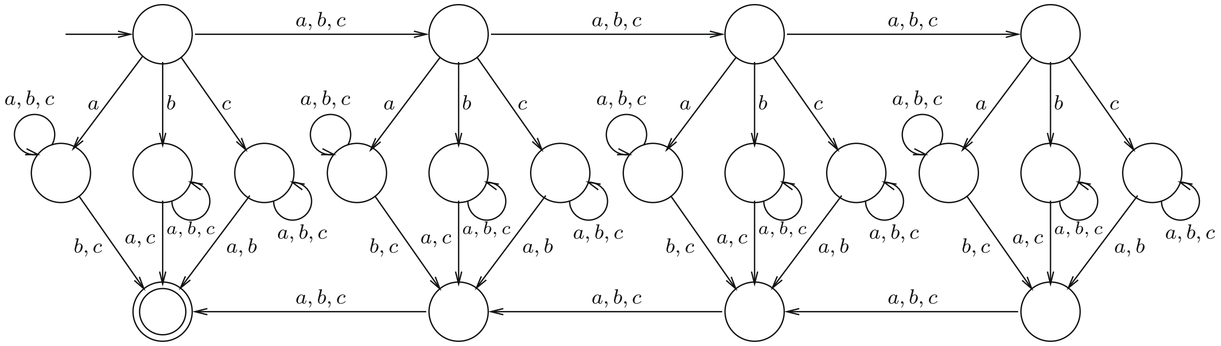

Fig. 1. Accepting non-palindromes over $\{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ for $s = 10$.

length $< 3n$, and so by Theorem 7, $L(M)$ is palindromic. If $L(A) \neq \emptyset$, then since $L(M'_{3n})$ contains only non-palindromes, we see that $L(M)$ is not palindromic.

We can determine if $L(A) = \emptyset$ efficiently by adding a new final state q_f and ϵ-transitions from all the final states of A to q_f, then performing a depth-first search to detect whether there are any paths from q_0 to q_f. This can be done in time linear in the number of states and transitions of A. If M has n states and t transitions, then A has $O(n^2)$ states and $O(tn)$ transitions. Hence we have proved the following theorem.

Theorem 8. *Let M be an NFA with n states and t transitions. The algorithm sketched above determines whether M accepts a palindromic language in $O(n^2 + tn)$ time.*

A different method runs slightly slower, but allows us to do a little more. We can mimic the construction for palindromes in Section 3, but adapt it for non-palindromes. Given an NFA M, we construct an NFA-ϵ M' that accepts the language

$$\left\{x \in \Sigma^* \ : \ \text{there exists } x' \in \Sigma^*, a \in \Sigma \text{ such that } |x| = |x'|, x \neq x'^R, \text{ and } xx' \in L(M) \text{ or } xax' \in L(M)\right\}.$$

The construction is similar to that in Lemma 2. On input x, we simulate M on xx' and xax' symbol-by-symbol, moving forward from the start state and backward from a final state. We need an additional boolean "flag" for each state to record whether or not we have processed a character in x' that would mismatch the corresponding character in x. If M has n states and t transitions, this construction produces an NFA-ϵ M' with $\leqslant 1 + 2n^2$ states and $O(t^2)$ transitions. From this we get, in analogy with Corollary 4, the following proposition.

Proposition 9. *Given an NFA M with n states and t transitions, we can determine in $O(n^2 + t^2)$ time if M accepts infinitely many non-palindromes.*

We now turn to the question of the optimality of the $3n$ bound given in Theorem 7. For an NFA over an alphabet of at least 2 symbols, the bound is indeed optimal, as the following example shows.

Proposition 10. *Let Σ be an alphabet of at least two symbols, containing the letters $\mathtt{a}$ and $\mathtt{b}$. For $n \geqslant 1$ define $L_n = (\mathtt{a}^{n-1}\Sigma)^*\mathtt{a}^{n-1}$. Then L_n can be accepted by an NFA with n states and a shortest non-palindrome in L_n is $\mathtt{a}^{n-1}\mathtt{a}\mathtt{a}^{n-1}\mathtt{b}\mathtt{a}^{n-1}$.*

Proof. The details are straightforward. □

For DFAs, however, the bound of $3n$ can be improved to $3n - 3$. To show this, we first prove the following lemma. A language L is called *slender* if there is a constant C such that, for all $n \geqslant 0$, the number of words of length n in L is less than C. The following characterization of slender regular languages has been independently rediscovered several times [20,30,26].

Theorem 11. *Let $L \subseteq \Sigma^*$ be a regular language. Then L is slender if and only if it can be written as a finite union of languages of the form uv^*w, where $u, v, w \in \Sigma^*$.*

Next we prove the following useful lemma concerning DFAs accepting slender languages.

Lemma 12. *Let L be a slender language accepted by a DFA M with n states, over an alphabet of two or more symbols. Then M must have a dead state.*

Proof. Without loss of generality, assume that every state of $M = (Q, \Sigma, \delta, q_0, F)$ is reachable from q_0, and that Σ contains the symbols a and b. We distinguish two cases:

1. M accepts a finite language. Consider the states reached from q_0 on a, a^2, $a^3, \ldots$. Eventually some state q must be repeated. This state q must be a dead state, for if not, M would accept an infinite language.
2. M accepts an infinite language. Then M has at least one *fruitful* cycle, that is, a cycle that produces infinitely many words in $L(M)$ as labels of paths starting at q_0, entering the cycle, going around the cycle some number of times, then exiting and eventually reaching a final state. Let C_1 be one fruitful cycle, and consider the following successful path involving C_1: $q_0 \xrightarrow{\alpha} q \xrightarrow{u} q \xrightarrow{\beta} f$, where $f \in F$ and the repetition of q denotes the cycle C_1, labeled with u. Without loss of generality assume the first letter of u is a. Since M is complete, denote $p = \delta(q, b)$.
   We claim that from p one cannot reach a fruitful cycle C_2. Indeed, let us assume the contrary; this means that there exists a successful path $q_0 \xrightarrow{\alpha} q \xrightarrow{u} q \xrightarrow{\gamma} r \xrightarrow{v} r \xrightarrow{\mu} f'$, with $f' \in F$ and the repetition of r denotes the cycle C_2 labeled with v. Let n be an arbitrary integer, and $0 \leqslant i \leqslant n$. There exist two integers k, l such that $k|u| = l|v| = m$. With this notation, observe that the words $\alpha u^{k(n-i)} \gamma v^{l(n+i)} \mu$ are all accepted by M and have the same length $2mn + |\alpha\gamma\mu|$. Since there are $n + 1$ such words, this proves that $L(M)$ has $\Omega(n)$ words of length n for large n—a contradiction.
   Thus, there exist a finite number of successful paths starting from p. However, considering the states reached from p by the words $a, a^2, a^3, \ldots$, one such state must repeat. This state is dead, for the alternative would contradict the finiteness of successful paths from p. □

Corollary 13. *If M is a DFA over an alphabet of at least two letters and $L(M)$ is palindromic, then M has a dead state.*

Proof. If $L(M)$ is palindromic, then by [15, Theorem 8] it can be written as a finite union of languages of the form $uv(tv)^*u^R$, where $u, v, t \in \Sigma^*$ and v, t are palindromes. By Theorem 11, this means $L(M)$ is slender. By Lemma 12, M has a dead state. □

We are now ready to prove the improved bound of $3n - 3$ for DFAs.

Theorem 14. *Let M be a DFA with n states. Then $L(M)$ is palindromic if and only if $\{x \in L(M) \ : \ |x| < 3n - 3\}$ is palindromic.*

Proof. One direction is clear.

If $M = (Q, \Sigma, \delta, q_0, F)$ is over a unary alphabet, then $L(M)$ is always palindromic, so the criterion is trivially true.

Otherwise M is over an alphabet of at least two letters. Assume $\{x \in L(M) \ : \ |x| < 3n - 3\}$ is palindromic. From Corollary 13, we see that M must have a dead state. But then we can delete such a dead state and all associated transitions, and all states reachable from the deleted dead state, to get a new NFA M' with at most $n - 1$ states that accepts the same language. We know from Theorem 7 that the palindromicity of $\{x \in L(M') \ : \ |x| < 3n - 3\}$ implies that M' is palindromic. □

Finally, we observe that $3n - 3$ is the best possible bound in the case of DFAs. To do so, we simply use the language L_n from Proposition 10 and observe it can be accepted by a DFA with $n + 1$ states; yet the shortest non-palindrome is of size $3n - 1$.

We end this section by noting that the related, but fundamentally different, problem of testing if $L = L^R$ was shown by Hunt [16] to be PSPACE-complete.

5. Testing if an NFA accepts a word matching a pattern

In this section, we consider the computational complexity of testing if an NFA accepts a word matching a given pattern. Specifically, we consider the following decision problem.

NFA PATTERN ACCEPTANCE

INSTANCE: An NFA M over the alphabet Σ and a pattern p over some alphabet Δ.

QUESTION: Does there exist $x \in \Sigma^+$ such that $x \in L(M)$ and x matches p?

Since the pattern p is given as part of the input, this problem is actually somewhat more general than the sort of problem formulated as question 1 of the introduction, where the language L was fixed.

We first consider the following result of Restivo and Salemi [27] (a more detailed proof appears in [4]). We give here a boolean matrix based proof (see Zhang [34] for a study of this boolean matrix approach to automata theory) that illustrates our general approach to the other problems treated in this section.

Theorem 15 (Restivo and Salemi). *Let L be a regular language and let Δ be an alphabet. The set P_Δ of all non-empty patterns $p \in \Delta^*$ such that p matches a word in L is effectively regular.*

Proof. Let $M = (Q, \Sigma, \delta, q_0, F)$ be an NFA such that $L(M) = L$. Suppose that $Q = \{0, 1, \ldots, n-1\}$. For $a \in \Sigma$, let B_a be the $n \times n$ boolean matrix whose (i,j) entry is 1 if $j \in \delta(i,a)$ and 0 otherwise. Let $\mathcal{B}$ denote the semigroup generated by the B_a's along with the identity matrix. For $w = w_0w_1 \cdots w_s$, where $w_i \in \Sigma$ for $i = 0, \ldots, s$, we write B_w to denote the matrix product $B_{w_0}B_{w_1} \cdots B_{w_s}$.

Without loss of generality, let $\Delta = \{1, 2, \ldots, k\}$. Observe that there exists a non-empty pattern $p = p_0p_1 \cdots p_r$, where $p_i \in \Delta$ for $i = 0, \ldots, r$, and a non-erasing morphism $h : \Delta^* \to \Sigma^*$ such that $h(p) \in L$ if and only if there exist k boolean matrices $B_1, \ldots, B_k \in \mathcal{B}$ such that $B_i = B_{h(i)}$ for $i \in \Delta$ and $B = B_{p_0}B_{p_1} \cdots B_{p_r}$ describes an accepting computation of M.

We construct an NFA $M' = (Q', \Delta, \delta', P, F')$ for P_Δ as follows. For simplicity, we permit M' to have multiple initial states, as specified by the set P. We define $Q' = \mathcal{B}^{k+1}$. The set P of initial states is given by $P = \mathcal{B}^k \times I$, where I denotes the identity matrix. In other words, the NFA M' uses the first k components of its state to record an initial guess of k boolean matrices $B_1, \ldots, B_k \in \mathcal{B}$. Let $[B_1, \ldots, B_k, A]$ denote some arbitrary state of M'. For $i = 1, \ldots, k$, the transition function δ' maps $[B_1, \ldots, B_k, A]$ to $[B_1, \ldots, B_k, AB_i]$. In other words, on input $p = p_0p_1 \cdots p_r \in \Delta^*$, M' uses the last component of its state to compute the product $B = B_{p_0}B_{p_1} \cdots B_{p_r}$. The set F' of final states of M' consists of all states of the form $[B_1, \ldots, B_k, B]$, where the matrix B contains a 1 in some entry $(0,j)$, where $j \in F$. In other words, M' accepts if and only if B describes an accepting computation of M. □

By consider unary patterns of the form a^k, we obtain the following corollary of Theorem 15.

Corollary 16. *Let $L \subseteq \Sigma^*$ be a regular language. The set of exponents k such that L contains a k-power is the union of a finite set with a finite union of arithmetic progressions. Further, this set of exponents is effectively computable.*

Observe that Theorem 15 implies the decidability of the **NFA PATTERN ACCEPTANCE** problem. We prove the following stronger result.

Theorem 17. *The* **NFA PATTERN ACCEPTANCE** *problem is PSPACE-complete.*

Proof. We first show that the problem is in PSPACE. By Savitch's theorem [29] it suffices to give an NPSPACE algorithm. Let $M = (Q, \Sigma, \delta, q_0, F)$, where $Q = \{0, 1, \ldots, n-1\}$. For $a \in \Sigma$, let B_a be the $n \times n$ boolean matrix whose (i,j) entry is 1 if $j \in \delta(i,a)$ and 0 otherwise. Let $\mathcal{B}$ denote the semigroup generated by the B_a's along with the identity matrix. For $w = w_0w_1 \cdots w_s \in \Sigma^*$, we write B_w to denote the matrix product $B_{w_0}B_{w_1} \cdots B_{w_s}$.

Let Δ be the set of letters occuring in p. We may suppose that $\Delta = \{1, 2, \ldots, k\}$. First, we non-deterministically guess k boolean matrices $B_1, \ldots, B_k$. Next, for each i, we verify that B_i is in the semigroup $\mathcal{B}$ by non-deterministically guessing a word $w = w_0w_1 \cdots w_s$ such that $B_i = B_w$. Since there are at most 2^{n^2} possible $n \times n$ boolean matrices, we may assume that $s \leqslant 2^{n^2}$. We thus guess w symbol-by-symbol and compute a sequence of matrices

$$B_{w_1}, B_{w_1w_2}, \ldots, B_{w_1w_2\cdots w_s},$$

reusing space after perfoming each matrix multiplication. We maintain an $O(n^2)$ bit counter to keep track of the length s of our guessed word w. If s exceeds 2^{n^2}, we reject on this branch of the non-deterministic computation.

Finally, if $p = p_0p_1 \cdots p_r$, we compute the matrix product $B = B_{p_0}B_{p_1} \cdots B_{p_r}$ and accept if and only if B describes an accepting computation of M.

To show hardness we reduce from the following PSPACE-complete problem [10, Problem AL6].

DFA INTERSECTION
INSTANCE: An integer $k \geqslant 1$ and k DFAs $A_1, A_2, \ldots, A_k$, each over the alphabet Σ.
QUESTION: Does there exist $x \in \Sigma^*$ such that x is accepted by each A_i, $1 \leqslant i \leqslant k$?

Let # be a symbol not in Σ. We construct, in linear time, a DFA M to accept the language

$$L(A_1)\,\#\,L(A_2)\,\#\,\cdots L(A_k)\,\#.$$

Any word in $L(M)$ matching the pattern a^k is of the form $(x\#)^k$. It follows that M accepts a word matching a^k if and only if there exists x such that $x \in L(A_i)$ for $1 \leqslant i \leqslant k$. This completes the reduction. □

We may define various variations or special cases of the **NFA PATTERN ACCEPTANCE** problem, such as: **NFA ACCEPTS A k-POWER**, **NFA ACCEPTS A $\geqslant k$-POWER**, **NFA ACCEPTS INFINITELY MANY k-POWERS**, **NFA ACCEPTS INFINITELY MANY $\geqslant k$-POWERS**, etc. We define and consider the computational complexity of these variations below.

NFA ACCEPTS A k-POWER.
INSTANCE: An NFA M over the alphabet Σ and an integer $k \geqslant 2$.
QUESTION: Does there exist $x \in \Sigma^+$ such that M accepts x^k?

NFA ACCEPTS A $\geqslant k$-POWER.
INSTANCE: An NFA M over the alphabet Σ.
QUESTION: Does there exist $x \in \Sigma^+$ and an integer $\ell \geqslant k$ such that M accepts x^ℓ?

The **NFA ACCEPTS A $\geqslant k$-POWER** problem is actually an infinite family of problems, each indexed by an integer $k \geqslant 2$. If k is fixed, the **NFA ACCEPTS A k-POWER** problem can be solved in polynomial time, as we now demonstrate.

Proposition 18. *Let M be an NFA with n states and t transitions, and set $N = n + t$, the size of M. For any fixed integer $k \geqslant 2$, there is an algorithm running in $O(n^{2k-1}t^k) = O(N^{2k-1})$ time to determine if M accepts a k-power.*

Proof. For a language $L \subseteq \Sigma^*$, we define

$$L^{1/k} = \left\{x \in \Sigma^* : x^k \in L\right\}.$$

Let $M = (Q, \Sigma, \delta, q_0, F)$ be an NFA with n states. We will construct an NFA-ϵ M' such that $L(M') = L(M)^{1/k}$. To determine whether or not M accepts a k-power, it suffices to check whether or not M' accepts a non-empty word.

The idea behind the construction of M' is as follows. On input x, M' first guesses $k - 1$ states $g_1, g_2, \ldots, g_{k-1} \in Q$ and then checks that

- $g_1 \in \delta(q_0, x)$,
- $g_{i+1} \in \delta(g_i, x)$ for $i = 1, 2, \ldots, k - 2$, and
- $\delta(g_{k-1}, x) \cap F \neq \emptyset$.

It is clear that such states $g_1, g_2, \ldots, g_{k-1}$ exist if and only if $x^k \in L(M)$.

Formally, the construction of M' is as follows. We define the NFA $M' = (Q', \Sigma, \delta', q_0', F')$ such that:

- $Q' = \{q_0'\} \cup Q^{2k-1}$. That is, except for q_0', each state of M' is a $(2k-1)$-tuple of the form $[g_1, g_2, \ldots, g_{k-1}, p_0, p_1, \ldots, p_{k-1}]$. The state g_i represents the i-th state guessed from M. The NFA M' will simulate in parallel the computations of M on input x starting from states $q_0, g_1, g_2, \ldots, g_{k-1}$, respectively. The state p_0 represents the current state of the simulation beginning from state q_0, and the states $p_1, p_2, \ldots, p_{k-1}$ represent the current states of the simulations beginning from states $g_1, g_2, \ldots, g_{k-1}$, respectively.
- q_0' is an additional state not in Q^{2k-1}. This state will have outgoing ϵ-transitions for each different combination of guesses g_i. The transition function on the start state is defined as

$$\delta'(q_0', \epsilon) = \{[g_1, g_2, \ldots, g_{k-1}, q_0, g_1, g_2, \ldots, g_{k-1}] : \forall\, i \in \{1, 2, \ldots, k-1\}, g_i \in Q\}.$$

- We define the transition function δ' on all other states as:

$$\delta'([g_1, g_2, \ldots, g_{k-1}, p_0, p_1, \ldots, p_{k-1}], a) = \left\{[g_1, g_2, \ldots, g_{k-1}, p_0', p_1', \ldots, p_{k-1}'] : \forall\, i \in \{0, 1, \ldots, k-1\}, p_i' \in \delta(p_i, a)\right\}$$

  for all $a \in \Sigma$.
- $F' = \{[g_1, g_2, \ldots, g_{k-1}, g_1, g_2, \ldots, g_{k-1}, t] : t \in F\}$. That is, we reach a state in F' on input x exactly when the guessed states g_i verify the conditions described above.

It should be clear from the construction that M' accepts $L(M)^{1/k}$. The number of states in M' is $n^{2k-1} + 1$, as, except for q_0', each state is a $(2k-1)$-tuple in which each coordinate can take on $|Q| = n$ possible values. For each state there are at most t^k distinct transitions. Testing whether or not $L(M')$ accepts a non-empty word can be done in linear time (since the only ϵ-transitions are transitions outgoing from q_0'), so the running time of our algorithm is $O(n^{2k-1}t^k)$. □

As before, we can use the same automaton to test if infinitely many k-powers are accepted.

Corollary 19. *We can decide if an NFA M with n states and t transitions accepts infinitely many k-powers in $O(n^{2k-1}t^k)$ time.*

If k is not fixed, we have the following result, which is an immediate consequence of Theorem 17 if k is given in unary. However, the problem remains in PSPACE even if k is given in binary, as we now demonstrate.

Theorem 20. *The problem* **NFA ACCEPTS A k-POWER** *is PSPACE-complete.*

Proof. We first show that the problem is in PSPACE. By Savitch's theorem [29] it suffices to give an NPSPACE algorithm. Let $M = (Q, \Sigma, \delta, q_0, F)$, where $Q = \{0, 1, \ldots, n-1\}$. For $a \in \Sigma$, let B_a be the $n \times n$ boolean matrix whose (i, j) entry is 1 if $j \in \delta(i, a)$ and 0 otherwise. Let $\mathcal{B}$ denote the semigroup generated by the B_a's.

We non-deterministically guess a boolean matrix B and verify that $B \in \mathcal{B}$ (i.e., $B = B_x$ for some $x \in \Sigma^*$), as illustrated in the proof of Theorem 17. Finally, we compute B_x^k efficiently by repeated squaring and verify that B_x^k contains a 1 in position (q_0, f) for some $f \in F$.

The proof for PSPACE-hardness is precisely that given in the proof of Theorem 17. □

Theorem 21. *For each integer $k \geqslant 2$, the problem* **NFA ACCEPTS A $\geqslant k$-POWER** *is PSPACE-complete.*

Proof. To show that the problem is in PSPACE, we use the same algorithm as in the proof of Theorem 20, with the following modification. In order to verify that M accepts an ℓ-power for some $\ell \geqslant k$, we first observe that by the same argument as in the proof of Proposition 45 below, if M accepts such an ℓ-power, then M accepts an ℓ-power for $k \leqslant \ell < k + n$. Thus, after non-deterministically computing B_x, we must compute B_x^ℓ for all $k \leqslant \ell < k + n$, and verify that at least one B_x^ℓ contains a 1 in position (q_0, f) for some $f \in F$.

To show PSPACE-hardness, we again reduce from the **DFA INTERSECTION** problem. Suppose that we are given r DFAs $A_1, A_2, \ldots, A_r$ and we wish to determine if the A_i's accept a common word x. We may suppose that $r \geqslant k$, since for any fixed k such a restriction does not affect the PSPACE-completeness of the **DFA INTERSECTION** problem. Let j be the smallest non-negative integer such that $r + j$ is prime. By Bertrand's Postulate [12, Theorem 418], we may take $j \leqslant r$. We now construct, in linear time, a DFA M to accept the language $L(A_1)\,\#\,L(A_2)\,\#\,\cdots L(A_r)\,\#(\Sigma^*\,\#)^j$. The DFA M accepts a $\geqslant k$-power if and only if it accepts an $(r + j)$-power. Moreover, M accepts an $(r + j)$-power if and only if there exists x such that $x \in L(A_i)$ for $1 \leqslant i \leqslant r$. This completes the reduction. □

In a similar fashion, we now show that the following decision problems are PSPACE-complete:

NFA ACCEPTS INFINITELY MANY k-POWERS.
INSTANCE: An NFA M over the alphabet Σ and an integer $k \geqslant 2$.
QUESTION: Does M accept x^k for infinitely many words x?

NFA ACCEPTS INFINITELY MANY $\geqslant k$-POWERS.
INSTANCE: An NFA M over the alphabet Σ.
QUESTION: Are there infinitely many pairs (x, i) such that $i \geqslant k$ and M accepts x^i?

Again, the **NFA ACCEPTS INFINITELY MANY $\geqslant k$-POWERS** problem is actually an infinite family of problems, each indexed by an integer $k \geqslant 2$. We will prove that these decision problems are PSPACE-complete by reducing from the following problem.

INFINITE CARDINALITY DFA INTERSECTION.
INSTANCE: An integer $k \geqslant 1$ and k DFAs $A_1, A_2, \ldots, A_k$, each over the alphabet Σ.
QUESTION: Do there exist infinitely many $x \in \Sigma^*$ such that x is accepted by each A_i, $1 \leqslant i \leqslant k$?

Lemma 22. *The decision problem* **INFINITE CARDINALITY DFA INTERSECTION** *is PSPACE-complete.*

Proof. First, let us see that the problem is in PSPACE. If the largest DFA has n states, then there is a DFA with at most n^k states that accepts $\bigcap_{1 \leqslant i \leqslant k} L(A_i)$. Now from Theorem 1(b), we know that there exist infinitely many x accepted by each A_i if and only if there is a word x of length ℓ, $n^k \leqslant \ell < 2n^k$, accepted by all the A_i. We can simply guess the symbols of x, ensuring with a counter that $n^k \leqslant |x| < 2n^k$, and checking by simulation that x is accepted by all the A_i. The counter uses at most $k \log n + \log 2$ bits, which is polynomial in the size of the input. This shows the problem is in non-deterministic polynomial space, and hence, by Savitch's theorem [29], in PSPACE.

Now,to see that **INFINITE CARDINALITY DFA INTERSECTION** is PSPACE-hard, we reduce from **DFA INTERSECTION**. For each DFA $A_i = (Q_i, \Sigma, \delta_i, q_{0,i}, F_i)$, we modify it to B_i as follows: we add a new initial state $q'_{0,i}$, and add the same transitions from it as from $q_{0,i}$. We then change all final states to non-final, and we make $q'_{0,i}$ final. We add a transition from all states that were previously final on a new letter ¢ (the same letter is used for each A_i), and a transition from all other states on ¢ to a new dead state d. Finally, we add transitions on all letters from d to itself. We claim B_i is a DFA and $L(B_i) = (L(A_i)¢)^*$. Furthermore, $\bigcap_{1 \leqslant i \leqslant k} L(A_i) \neq \emptyset$ if and only if $\bigcap_{1 \leqslant i \leqslant k} L(B_i)$ is infinite.

Suppose $\bigcap_{1 \leqslant i \leqslant k} L(A_i) \neq \emptyset$. Then there exists x accepted by each of the A_i. Then $(x¢)^*$ is accepted by each of the B_i, so $\bigcap_{1 \leqslant i \leqslant k} L(B_i)$ is infinite.

Now suppose $\bigcap_{1 \leqslant i \leqslant k} L(B_i)$ is infinite. Choose any non-empty $x \in \bigcap_{1 \leqslant i \leqslant k} L(B_i) = \bigcap_{1 \leqslant i \leqslant k} (L(A_i)¢)^*$. Thus x must be of the form $y_1¢y_2¢ \cdots y_j¢$ for some $j \geqslant 1$, where each y_i is accepted by all the A_i. Hence, in particular, y_1 is accepted by all the A_i, and so $\bigcap_{1 \leqslant i \leqslant k} L(A_i) \neq \emptyset$. □

We are now ready to prove

Theorem 23. *The decision problem* **NFA ACCEPTS INFINITELY MANY k-POWERS** *is PSPACE-complete.*

Proof. First, let us see that the problem is in PSPACE. We claim that an NFA M with n states accepts infinitely many k-powers if and only if it accepts a k-power x^k with $2^{n^2} \leqslant |x| < 2^{n^2+1}$.

One direction is clear. For the other direction, we use boolean matrices, as in the proof of Theorem 20. We can construct a DFA $M' = (Q', \Sigma, \delta', q_0', F')$ of 2^{n^2} states that accepts $L^{1/k} = \{x \in \Sigma^* \ : \ x^k \in L(M)\}$, as follows: the states are $n \times n$ boolean matrices. The initial state q_0' is the identity matrix. If B_a is the boolean matrix with a 1 in entry (i, j) if $j \in \delta(q_i, a)$ and 0 otherwise, then $\delta'(B, a) = BB_a$. The set of final states is $F' = \{B \ : \ \text{the } (0, j) \text{ entry of } B^k \text{ is 1 for some } q_j \in F\}$.

The idea of this construction is that if $x = a_1a_2 \cdots a_i$, then $\delta(q_0', x) = B_{a_1} \cdots B_{a_i}$. Now we use Theorem 1(b) to conclude that M' accepts infinitely many words if and only if it accepts a word x with $2^{n^2} \leqslant |x| < 2^{n^2+1}$. But $L(M') = L(M)^{1/k}$.

Thus, to check if M accepts infinitely many k-powers, we simply guess the symbols of x, stopping when $2^{n^2} \leqslant |x| < 2^{n^2+1}$, and verify that M accepts x^k. We can do this by accumulating $B_{a_1} \cdots B_{a_k}$ and raising the result to the k-th power, as before. We need $n^2 + 1$ bits to keep track of the counter, so the result is in NPSPACE, and hence in PSPACE.

Now we argue that **NFA ACCEPTS INFINITELY MANY k-POWERS** is PSPACE-hard. To do so, we reduce from **INFINITE CARDINALITY DFA INTERSECTION**. Given DFAs $A_1, A_2, \ldots, A_k$, we can easily construct a DFA A to accept $L(A_1)\# \cdots L(A_k)\#$. Clearly A accepts infinitely many k-powers if and only if $\bigcap_{1 \leqslant i \leqslant k} L(A_i)$ is infinite. □

Theorem 24. *For each integer $k \geqslant 2$, the problem* **NFA ACCEPTS INFINITELY MANY $\geqslant k$-POWERS** *is PSPACE-complete.*

Proof. Left to the reader. □

We conclude by observing that all of the problems that we have shown in this section to be PSPACE-complete remain PSPACE-complete even when the input automaton is a DFA rather than an NFA; this is evident from the proofs given above.

6. Testing if an NFA accepts a non-k-power

In the previous section, we showed that it is computationally hard to test if an NFA accepts a k-power (when k is not fixed). In this section, we show how to test if an NFA accepts a non-k-power. Again, we find it more congenial to discuss the opposite problem, which is whether an NFA accepts nothing but k-powers.

First, we need several classical results from the theory of combinatorics on words. The following theorem is due to Lyndon and Schützenberger [21].

Theorem 25. *If x, y, and z are words satisfying an equation $x^iy^j = z^k$, where $i, j, k \geqslant 2$, then they are all powers of a common word.*

The next result is also due to Lyndon and Schützenberger.

Theorem 26. *Let u and v be non-empty words. If $uv = vu$, then there exists a word x and integers $i, j \geqslant 1$, such that $u = x^i$ and $v = x^j$. In other words, u and v are powers of a common word.*

The following result can be derived from Theorem 26.

Corollary 27. *Let u and v be non-empty words. If $u^r = v^s$ for some $r, s \geqslant 1$, then u and v are powers of a common word.*

Ito et al. [17] gave a proof of the next proposition.

Proposition 28. *Let u and v be non-empty words. If u and v are not powers of a common word, then for any integers $r, s \geqslant 1$, $r \neq s$, at least one of u^rv or u^sv is primitive.*

The next result is due to Shyr and Yu [32].

Theorem 29. *Let p and q be primitive words, $p \neq q$. The set p^+q^+ contains at most one non-primitive word.*

Next we prove the following analogue of Theorem 7, from which we will derive an efficient algorithm for testing if a finite automaton accepts only k-powers.

Theorem 30. *Let L be accepted by an n-state NFA M and let $k \geqslant 2$ be an integer.*

1. *Every word in L is a k-power if and only if every word in the set $\{x \in L : |x| \leqslant 3n\}$ is a k-power.*
2. *All but finitely many words in L are k-powers if and only if every word in the set $\{x \in L : n \leqslant |x| \leqslant 3n\}$ is a k-power.*

Further, if M is a DFA over an alphabet of size $\geqslant 2$, then the bound $3n$ may be replaced by $3n - 3$.

Ito et al. [17] proved a similar result for primitive words: namely, that if L is accepted by an n-state DFA over an alphabet of two or more letters and contains a primitive word, then it contains a primitive word of length $\leqslant 3n - 3$. In other words, every word in L is a power if and only if every word in the set $\{x \in L : |x| \leqslant 3n - 3\}$ is a power. However, this result does not imply Theorem 30, as one can easily construct a regular language L where every word in L that is not a k-power is nevertheless non-primitive: for example, $L = \{a^{k+1}\}$.

We shall use the next result to characterize those regular languages consisting only of k-powers.

Proposition 31. *Let u, v, and w be words, $v \neq \epsilon$, $uw \neq \epsilon$, and let $f, g \geqslant 1$ be integers, $f \neq g$. If $uv^f w$ and $uv^g w$ are non-primitive, then $uv^n w$ is non-primitive for all integers $n \geqslant 1$. Further, if uvw and uv^2w are k-powers for some integer $k \geqslant 2$, then v and $uv^n w$ are k-powers for all integers $n \geqslant 1$.*

Proof. Suppose $uv^f w$ and $uv^g w$ are non-primitive. Then $v^f wu$ and $v^g wu$ are non-primitive. Let x and y be the primitive roots of v and wu, respectively, so that $v = x^i$ and $wu = y^j$ for some integers $i, j \geqslant 1$. If $x \neq y$, then by Proposition 28, one concludes that at least one of $v^f wu$ or $v^g wu$ is primitive, a contradiction.

If $x = y$, then for all integers $n \geqslant 1$, $v^n wu = x^{ni+j}$ is clearly non-primitive, and consequently, $uv^n w$ is non-primitive, as required. Let us now suppose that uvw and uv^2w are k-powers for some $k \geqslant 2$. Then $vwu = x^{i+j}$ and $v^2wu = x^{2i+j}$ are both k-powers as well. We claim that the following must hold:

$$\begin{aligned} i + j &\equiv 0 \pmod{k} \\ 2i + j &\equiv 0 \pmod{k}. \end{aligned}$$

To see this, write $vwu = z^k$ for some word z. Then $z^k = x^{i+j}$, so by Corollary 27 z and x are powers of a common word. Since x is primitive it follows that z is a power of x. In particular, $|x|$ divides $|z|$ and $i + j$ is a multiple of k, as claimed. A similar argument applies to v^2wu.

We conclude that $i \equiv j \equiv 0 \pmod{k}$, and hence, $v = x^i$ is a k-power. Moreover, $v^n wu = x^{ni+j}$ is also a k-power for all integers $n \geqslant 1$, and consequently, $uv^n w$ is a k-power, as required. □

The characterization due to Ito et al. [17, Proposition 10] (see also Dömösi et al. [7, Theorem 3]) of the regular languages consisting only of powers, along with Theorem 11, implies that any such language is slender. A simple application of the Myhill–Nerode theorem gives the following weaker result.

Proposition 32. *Let L be a regular language and let $k \geqslant 2$ be an integer. If all but finitely many words of L are k-powers, then L is slender. In particular, if L is accepted by an n-state DFA and all words in L of length $\geqslant \ell$ are k-powers, then for all $r \geqslant \ell$, the number of words in L of length r is at most n.*

Proof. Let x^k and y^k be distinct words in L of length $r \geqslant \ell$. Then x and y are inequivalent with respect to the Myhill–Nerode equivalence relation, since $y^k \in L$ but $xy^{k-1} \notin L$. The Myhill–Nerode equivalence relation on L thus has index at least as large as the number of distinct words of length r in L. Since the index of the Myhill–Nerode relation is at most n, it follows that there is a bounded number of words of length r in L, so that L is slender, as required. □

The following characterization is analogous to the characterization of palindromic regular languages given in [15, Theorem 8].

Theorem 33. *Let $L \subseteq \Sigma^*$ be a regular language and let $k \geqslant 2$ be an integer. The language L consists only of k-powers if and only if it can be written as a finite union of languages of the form uv^*w, where $u, v, w \in \Sigma^*$ satisfy the following: there exists a primitive word $x \in \Sigma^*$ and integers $i, j \geqslant 0$ such that $v = x^{ik}$ and $wu = x^{jk}$.*

Proof. The "if" direction is clear; we prove the "only if" direction. Let L consist only of k-powers. Then by Proposition 32, L is slender. By Theorem 11, L can be written as a finite union of languages of the form uv^*w. By examining the proof of Proposition 31, one concludes that u, v, and w have the desired properties. □

We shall need the following lemma for the proof of Theorem 30.

Lemma 34. *Let L be a regular language accepted by an n-state NFA M and let $k \geqslant 2$ be an integer. If L contains a non-k-power of length $\geqslant n$, then L contains infinitely many non-k-powers.*

Proof. Let $s \in L$ be a non-k-power such that $|s| \geqslant n$. Consider an accepting computation of M on s. Such a computation must contain at least one repeated state. It follows that there exists a decomposition $s = uvw$, $v \neq \epsilon$, such that $uv^*w \subseteq L$. Let x be the primitive root of v, so that $v = x^i$ for some positive integer i.

Suppose that $wu = \epsilon$. Since $s = v = x^i$ is not a k-power, it follows that $i \not\equiv 0 \pmod{k}$. Moreover, there exist infinitely many positive integers ℓ such that $\ell i \not\equiv 0 \pmod{k}$, and so by Corollary 27, there exist infinitely many words of the form $v^\ell = x^{\ell i}$ that are non-k-powers in L, as required.

Suppose then that $wu \neq \epsilon$. Let y be the primitive root of wu, so that $wu = y^j$ for some positive integer j. We have two cases.

Case 1: $x = y$. Since uvw is a not a k-power, vwu is also not a k-power, and thus we have $i + j \not\equiv 0 \pmod{k}$. Moreover, there are infinitely many positive integers ℓ such that $\ell i + j \not\equiv 0 \pmod{k}$. For all such ℓ, the word $v^\ell wu = x^{\ell i+j}$ is not a k-power, and hence the word $uv^\ell w$ is a non-k-power in L. We thus have infinitely many non-k-powers in L, as required.

Case 2: $x \neq y$. By Theorem 29, v^*wu contains infinitely many primitive words. Thus, uv^*w contains infinitely many non-k-powers, as required. □

We are now ready to prove Theorem 30.

Proof (*of Theorem 30*). The proof is similar to that of [17, Proposition 7]. It suffices to prove statement (2) of the theorem, since statement (1) follows immediately from (2) and Lemma 34.

Suppose that L contains infinitely many non-k-powers. Then L contains a non-k-power s with $|s| \geqslant n$. Suppose, contrary to statement (2), that a shortest such s has $|s| > 3n$. Then any computation of M on s must repeat some state at least four times. It follows that there exists a decomposition $s = uv_1v_2v_3w$, $v_1, v_2, v_3 \neq \epsilon$, such that $uv_1^*v_2^*v_3^*w \subseteq L$. We may assume further that $|v_1v_2v_3| \leqslant 3n$, so that $wu \neq \epsilon$.

Let p_1, p_2, p_3, and q be the primitive roots of v_1, v_2, v_3, and wu, respectively. Let $v_1 = p_1^{i_1}$, $v_2 = p_2^{i_2}$, $v_3 = p_3^{i_3}$, and $wu = q^j$, for some integers $i_1, i_2, i_3, j > 0$. We consider three cases.

Case 1: $p_1 = p_2 = p_3 = q$. Without loss of generality, suppose that $|v_1| \leqslant |v_2| \leqslant |v_3|$. Since $|s| > 3n$, we must have $|uv_3w| \geqslant n$, and thus $|uv_1v_3w| \geqslant n$ and $|uv_2v_3w| \geqslant n$. By assumption, the words $v_3wu = q^{i_3+j}$, $v_1v_3wu = q^{i_1+i_3+j}$, and $v_2v_3wu = q^{i_2+i_3+j}$ are k-powers, whereas the word $v_1v_2v_3wu = q^{i_1+i_2+i_3+j}$ is not. Applying Corollary 27, we deduce that the following system of equations

$$\begin{aligned} i_1 + i_2 + i_3 + j &\not\equiv 0 \pmod{k} \\ i_3 + j &\equiv 0 \pmod{k} \\ i_1 + i_3 + j &\equiv 0 \pmod{k} \\ i_2 + i_3 + j &\equiv 0 \pmod{k} \end{aligned}$$

must be satisfied. However, it is easy to see that this is impossible.

Case 2: $p_1 \neq q$ and $p_2 = p_3 = q$. If $|v_1wu| \leqslant n$, then let ℓ be the smallest positive integer such that $n \leqslant |v_1^\ell wu| < |v_1^{\ell+1}wu| \leqslant |s|$. Then by Proposition 28, one of the words $v_1^\ell wu$ or $v_1^{\ell+1}wu$ is primitive. Hence, at least one of the words $uv_1^\ell w$ or $uv_1^{\ell+1}w$ is a primitive word in L, contradicting the minimality of s.

If, instead, $|v_1wu| > n$, then we have $n < |v_1wu| < |v_1v_2wu| \leqslant |s|$. Again, by Proposition 28, one of the words v_1wu or v_1v_2wu is primitive. Hence, at least one of the words uv_1w or uv_1v_2w is a primitive word in L, contradicting the minimality of s.

Case 3: $p_1 \neq q$ and $p_2 \neq q$. In this case we choose the smaller of v_1 and v_2 to "pump", so without loss of generality, suppose $|v_1| \leqslant |v_2|$. Let ℓ be the smallest positive integer such that $n \leqslant |v_1^\ell wu| < |v_1^{\ell+1}wu| \leqslant |s|$. Note that $|v_1^2wu| \leqslant |v_1v_2wu| < |s|$, so such an ℓ must exist. Then by Proposition 28, one of the words $v_1^\ell wu$ or $v_1^{\ell+1}wu$ is primitive. Hence, at least one of the words $uv_1^\ell w$ or $uv_1^{\ell+1}w$ is a primitive word in L, contradicting the minimality of s.

All remaining possibilities are symmetric to the cases considered above. Since in all cases we derive a contradiction, it follows that if L contains infinitely many non-k-powers, it contains a non-k-power s, where $n \leqslant |s| \leqslant 3n$.

It remains to consider the situation where M is a DFA over an alphabet of size $\geqslant 2$. Let $a \neq b$ be alphabet symbols of M. If M does not have a dead state, then for every integer $i \geqslant n-1$, there exists a word x, $|x| \leqslant n-1$, such that $a^ibx \in L$. These words a^ibx are all distinct and primitive. Thus, whenever M has no dead state, M always accepts infinitely many non-k-powers, and, in particular, M accepts a non-k-power s, where $n \leqslant |s| \leqslant 2n-1$.

If, on the other hand, M does have a dead state, then we may delete this dead state and apply the earlier argument with the bound $3n-3$ in place of $3n$.

Finally, the converse of statement (2) follows immediately from Lemma 34. □

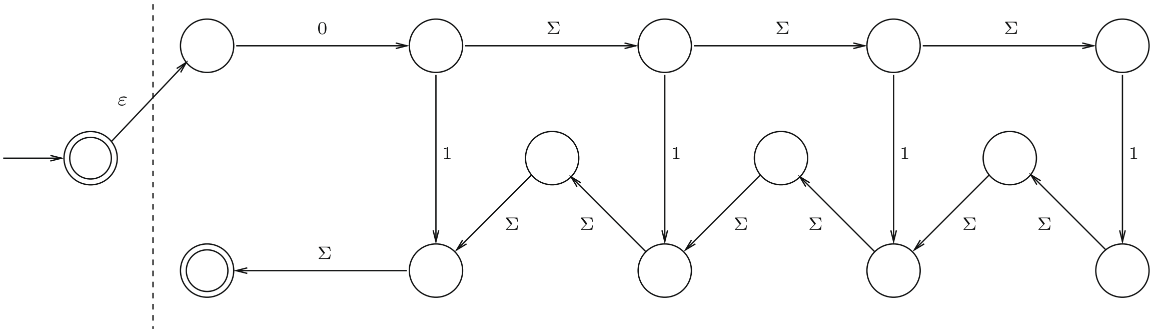

Fig. 2. One lobe of the NFA for $k = 3, r = 12$ and 0, 1 conflicting symbols.

We can now deduce the following algorithmic result.

Theorem 35. *Let $k \geqslant 2$ be an integer. Given an NFA M with n states and t transitions, it is possible to determine if every word in $L(M)$ is a k-power in $O(n^3 + tn^2)$ time.*

Proof. The proof is exactly analogous to that of Theorem 8, and we only indicate what needs to be changed. Suppose M has t states. We create an NFA, M'_r, for $r = 3t$, such that no word in $L(M'_r)$ is a k-power, and M'_r accepts all non-k-powers of length $\leqslant r$ (and perhaps some other non-k-powers).

Note that we may assume that $k \leqslant r$. If $k > r$, then no word of length $\leqslant r$ is a k-power. In this case, to obtain the desired answer it suffices to test if the set $\{x \in L(M) : |x| \leqslant r\}$ is empty. However, this set is empty if and only if $L(M)$ is empty, and this is easily verified in linear time.

We now form a new NFA A as the cross-product of M'_r with M. From Theorem 30, it follows that $L(A) = \emptyset$ iff every word in $L(M)$ is a k-power. We can determine if $L(A) = \emptyset$ by checking (using depth-first search) whether any final states of A are reachable from the start state.

It remains to see how M'_r is constructed. If the length of a word x accepted by M'_r is a multiple of k, x can be partitioned into k sections of equal length. In order for M'_r to accept x, the NFA must 'verify' a symbol mismatch between two symbols found in different sections but in the same position.

If x is a non-k-power, then a symbol mismatch will occur between two sections of x, call them s_i and s_j. This means that s_i and s_j differ in at least one position. Comparing s_i and s_j to s_1, the first section of x, we notice that at least one of s_i or s_j must have a symbol mismatch with s_1 (otherwise $s_1 = s_i = s_j$, which would give a contradiction). Therefore, when checking x for a symbol mismatch, it is sufficient to only check s_1 against each of the remaining $k - 1$ sections, as opposed to checking all $\binom{k}{2}$ possibilities.

In order to construct M'_r, we create a series of 'lobes', each of which is connected to the start state by an ϵ-transition. Each lobe represents three simultaneous 'guesses' made by the NFA, which are:

- Which alphabet symbols will conflict and in which order. The number of possible conflict pairs is $|\Sigma|\,(|\Sigma| - 1)$.
- The section in which there will be a symbol mismatch with the first section. There are $k - 1$ possible sections.
- The position in which the conflict will occur. In the worst case when the length of the input is r, there will be at most r/k possible positions.

This gives a total of at most $|\Sigma|\,(|\Sigma| - 1) \cdot (k - 1) \cdot r/k$ lobes. The construction of each lobe is illustrated in Fig. 2.

Each lobe contains at most $r + 1$ states. In addition to these lobes, we also require a k-state submachine to accept all words whose lengths are not a multiple of k.

In total, M'_r has at most

$$|\Sigma|\,(|\Sigma| - 1) \cdot (k - 1) \cdot \frac{r}{k} \cdot (r + 1) + k + 1 \in O(r^2)$$

states (since $k \leqslant r$), and similarly, $O(r^2)$ transitions. After constructing the cross-product, this gives a $O(n^3 + tn^2)$ bound on the time required to determine if every word in $L(M)$ is a k-power. □

Theorem 30 suggests the following question: if M is an NFA with n states that accepts at least one non-k-power, how long can a shortest non-k-power be? Theorem 30 proves an upper bound of $3n$. A lower bound of $2n - 1$ for infinitely many n follows easily from the obvious $(n + 1)$-state NFA accepting $\mathtt{a}^n(\mathtt{a}^{n+1})^*$, where n is divisible by k. However, Ito et al. [17] gave a very interesting example that improves this lower bound: if $x = ((ab)^n a)^2$ and $y = baxab$, then x and xyx are squares, but $xyxyx$ is not a power. Hence, the obvious $(8n+8)$-state NFA that accepts $x(yx)^*$ has the property that the shortest non-k-power accepted is of length $20n+18$. This improves the lower bound for infinitely many n.

We now generalize their lower bound.

Proposition 36. *Let $k \geqslant 2$ be fixed. There exist infinitely many NFAs M with the property that if M has r states, then the shortest non-k-power accepted is of length $\left(2+\frac{1}{2k-2}\right)r - O(1)$.*

Proof. Let $u=(ab)^n a$, $x=u^k$, and $y=x^{-1}(xbau^{-1}x)^k x^{-1}$. Thus $xyx=(xbau^{-1}x)^k$. Hence x and xyx are both k-powers.

However, $xyxyx$ is not a k-power. To see this, assume it is, and write $xyxyx=g_1g_2\cdots g_k$. Look at the character in position $2kn-2n+k$ (indexing beginning with 1) in g_1 and g_k. In g_1 it is a, and in g_k it is b, so $xyxyx$ is not a k-power.

We can accept $x(yx)^*$ with an NFA using $|xy|$ states. The shortest non-k-power is $xyxyx$, which is of length m.

We have $|u|=2n+1$, $|x|=k(2n+1)$, $|y|=k(4kn-6n+2k-1)$, $r=|xy|=2k(2kn-2n+k)$, and $m=|xyxyx|=k(8kn-6n+4k+1)$. Thus $m=\frac{4k-3}{2k-2}r-\frac{k}{k-1}=\left(2+\frac{1}{2k-2}\right)r-O(1)$. □

Next, we apply part (2) of Theorem 30 to obtain an algorithm to check if an NFA accepts infinitely many non-k-powers.

Theorem 37. *Let $k \geqslant 2$ be an integer. Given an NFA M with n states and t transitions, it is possible to determine if all but finitely many words in $L(M)$ are k-powers in $O(n^3+tn^2)$ time.*

Proof. The proof is similar to that of Theorem 35. The only difference is that in view of part (2) of Theorem 30 we instead construct M_r' to accept all non-k-powers s, where $n \leqslant |s| \leqslant 3n$. We leave the details to the reader. □

7. Automata accepting only powers

In this section, we move from the problem of testing if an automaton accepts only k-powers to the problem of testing if it accepts only powers (of any kind). Just as Theorem 30 was the starting point for our algorithmic results in Section 6, the following theorem of Ito et al. [17] is the starting point for our algorithmic results in this section. We state the theorem in a stronger form than was originally presented by Ito et al.

Theorem 38. *Let L be accepted by an n-state NFA M.*

1. *Every word in L is a power if and only if every word in the set $\{x \in L : |x| \leqslant 3n\}$ is a power.*
2. *All but finitely many words in L are powers if and only if every word in the set $\{x \in L : n \leqslant |x| \leqslant 3n\}$ is a power.*

Further, if M is a DFA over an alphabet of size $\geqslant 2$, then the bound $3n$ may be replaced by $3n-3$.

We next prove an analogue of Proposition 32. We need the following result, first proved by Birget [3], and later, independently, in a weaker form, by Glaister and Shallit [11].

Theorem 39. *Let $L \subseteq \Sigma^*$ be a regular language. Suppose there exists a set of pairs*

$$S=\{(x_i,y_i)\in \Sigma^* \times \Sigma^* : 1 \leqslant i \leqslant n\}$$

such that

- *$x_iy_i \in L$ for $1 \leqslant i \leqslant n$ and*
- *either $x_iy_j \notin L$ or $x_jy_i \notin L$ for $1 \leqslant i,j \leqslant n$, $i \neq j$.*

Then any NFA accepting L has at least n states.

Proposition 40. *Let M be an n-state NFA and let ℓ be a non-negative integer such that every word in $L(M)$ of length $\geqslant \ell$ is a power. For all $r \geqslant \ell$, the number of words in $L(M)$ of length r is at most 7n.*

Proof. Let $r \geqslant \ell$ be an arbitrary integer. The proof consists of three steps:

Step 1. We consider the set A of words w in $L(M)$ such that $|w|=r$ and w is a k-power for some $k \geqslant 4$. For each such w, write $w=x^i$, where x is a primitive word, and define a pair (x^2,x^{i-2}). Let S_A denote the set of such pairs. Consider two pairs in S_A: (x^2,x^{i-2}) and (y^2,y^{j-2}). The word x^2y^{j-2} is primitive by Theorem 25 and hence is not in $L(M)$. The set S_A thus satifies the conditions of Theorem 39. Since $L(M)$ is accepted by an n-state NFA, we must have $|S_A| \leqslant n$ and thus $|A| \leqslant n$.

Step 2. Next we consider the set B of cubes of length r in $L(M)$. For each such cube $w=x^3$, we define a pair (x,x^2). Let S_B denote the set of such pairs. Consider two pairs in S_B: (x,x^2) and (y,y^2). Suppose that xy^2 and yx^2 are both in $L(M)$.

The word xy^2 is certainly not a cube; we claim that it cannot be a square. Suppose it were. Then $|x|$ and $|y|$ are even, so we can write $x = x_1x_2$ and $y = y_1y_2$ where $|x_1| = |x_2| = |y_1| = |y_2|$. Now if $xy^2 = x_1x_2y_1y_2y_1y_2$ is a square, then $x_1x_2y_1 = y_2y_1y_2$, and so $y_1 = y_2$. Thus y is a square; write $y = z^2$. By Theorem 25, $yx^2 = z^2x^2$ is primitive, contradicting our assumption that $yx^2 \in L(M)$. It must be the case then that xy^2 is a k-power for some $k \geqslant 4$. Thus, $xy^2 = u^k$ for some primitive u uniquely determined by x and y. With each pair of cubes x^3 and y^3 such that both xy^2 and yx^2 are in $L(M)$ we may therefore associate a k-power $u^k \in L(M)$ of length r, where $k \geqslant 4$. We have already established in Step 1 that the number of such k-powers is at most n. It follows that by deleting at most n pairs from the set S_B we obtain a set of pairs satisfying the conditions of Theorem 39. We must therefore have $|S_B| \leqslant 2n$ and thus $|B| \leqslant 2n$.

Step 3. Finally we consider the set C of squares of length r in $L(M)$. For each such square $w = x^2$, we define a pair (x, x). Let S_C denote the set of such pairs. Consider two pairs in S_C: (x, x) and (y, y). Suppose that xy and yx are both in $L(M)$. The word xy is not a square and must therefore be a k-power for some $k \geqslant 3$. We write $xy = u^k$ for some primitive u uniquely determined by x and y. In Steps 1 and 2 we established that the number of k-powers of length r, $k \geqslant 3$, is $|A| + |B| \leqslant 3n$. It follows that by deleting at most $3n$ pairs from the set S_C we obtain a set of pairs satisfying the conditions of Theorem 39. We must therefore have $|S_C| \leqslant 4n$ and thus $|C| \leqslant 4n$.

Putting everything together, we see that there are $|A| + |B| + |C| \leqslant 7n$ words of length r in $L(M)$, as required. □

The bound of $7n$ in Proposition 40 is almost certainly not optimal.

We now prove the following algorithmic result.

Theorem 41. *Given an NFA M with n states, it is possible to determine if every word in $L(M)$ is a power in $O(n^5)$ time.*

Proof. First, we observe that we can test whether a word w of length n is a power in $O(n)$ time, using a linear-time string matching algorithm, such as Knuth–Morris–Pratt [19]. To do so, search for $w = a_1a_2 \cdots a_n$ in the word $x = a_2 \cdots a_na_1 \cdots a_{n-1}$. Then w appears in x iff w is a power. Furthermore, if the leftmost occurrence of w in x appears beginning at a_i, then w is a $n/(i-1)$ power, and this is the largest exponent of a power that w is.

Now, using Theorem 38, it suffices to test all words in $L(M)$ of length $\leqslant 3n$; every word in $L(M)$ is a power iff all of these words are powers. On the other hand, by Proposition 40, if all words are powers, then the number of words of each length is bounded by $7n$. Thus, it suffices to enumerate the words in $L(M)$ of lengths $1, 2, \ldots, 3n$, stopping if the number of such words in any length exceeds $7n$. If all these words are powers, then every word is a power. Otherwise, if we find a non-power, or if the number of words in any length exceeds $7n$, then not every word is a power.

By the work of Mäkinen [22] or Ackerman and Shallit [1], we can enumerate these words in $O(n^5)$ time. □

Using part (2) of Theorem 38 along with Proposition 40, we can prove the following.

Theorem 42. *Given an NFA M with n states, we can decide if all but finitely many words in $L(M)$ are non-powers in $O(n^5)$ time.*

Proof. The proof is analogous to that of Theorem 41. The only difference is that here we need only enumerate the words in $L(M)$ of lengths $n, n+1, \ldots, 3n$. □

8. Bounding the length of a smallest power

In Section 6 we gave an upper bound on the length of a smallest non-k-power accepted by an n state NFA. In this section, we study the complementary problem of bounding the length of the smallest k-power accepted by an n-state NFA.

Proposition 43. *Let M be an NFA with n states and let $k \geqslant 2$ be an integer. If $L(M)$ contains a k-power, then $L(M)$ contains a k-power of length $\leqslant kn^k$.*

Proof. Consider the NFA-ϵ M' accepting $L(M)^{1/k}$ defined in the proof of Proposition 18. The only transitions from the start state of M' are ϵ-transitions to submachines whose states are $(2k-1)$-tuples of the form $[g_1, g_2, \ldots, g_{k-1}, p_0, p_1, \ldots, p_{k-1}]$, where the first $(k-1)$-elements of the tuple are fixed. Thus we may consider $L(M')$ as a finite union of languages, each accepted by an NFA of size n^k. It follows that if M' accepts a non-empty word w, it accepts such a w of length $\leqslant n^k$. However, M' accepts w if and only if M accepts w^k. We conclude that if $L(M)$ contains a k-power, it contains one of length $\leqslant kn^k$. □

We now give a lower bound on the size of the smallest k-power accepted by an n-state DFA.

Proposition 44. *Let $k \geqslant 2$ be an integer. There exist infinitely many DFAs M_n such that*

(a) *M_n has $O(kn)$ states.*
(b) *The shortest k-power accepted by M_n is of length $k \cdot \Omega\left(\binom{n}{k}\right)$.*

Proof. For $n \geqslant k$, let

$$L_n = (\mathtt{a}^n)^+\mathtt{b}(\mathtt{a}^{n-1})^+\mathtt{b}\cdots(\mathtt{a}^{n-k+1})^+\mathtt{b}.$$

Then L_n is accepted by a DFA with $O(kn)$ states, and the shortest k-power in L_n is $(\mathtt{a}^\ell\mathtt{b})^k$, where

$$\ell = \mathrm{lcm}(n, n-1, \ldots, n-k+1) \geqslant n(n-1)\cdots(n-k+1)/k! = \binom{n}{k},$$

as required. □

Next we consider the length of a smallest power (rather than k-power).

Proposition 45. *Let M be an NFA with n states. If $L(M)$ contains a power, it contains a k-power for some k, $2 \leqslant k \leqslant n+1$.*

Proof. Suppose to the contrary that the smallest k for which $L(M)$ contains a k-power w^k satisfies $k > n+1$. For some accepting computation of M on w^k let $q_1, q_2, \ldots, q_{k-1}$ be the states reached by M after reading $w, w^2, \ldots, w^{k-1}$, respectively. Since $k > n+1$, there exist i and j where $1 \leqslant i < j \leqslant k-1$ and $q_i = q_j$. It follows that M accepts w^ℓ for some ℓ, $2 \leqslant \ell < k$, contradicting the minimality of k. We conclude that if $L(M)$ contains a k-power, we may take $k \leqslant n+1$. □

Proposition 46. *Let M be an NFA with n states. If $L(M)$ contains a power, then $L(M)$ contains a power of length $\leqslant (n+1)n^{n+1}$.*

Proof. Apply Propositions 45 and 43. □

We now give a lower bound.

Proposition 47. *There exist infinitely many DFAs M_n such that*

- *M_n has $O(n)$ states.*
- *The shortest power accepted by M_n is of length $e^{\Omega(\sqrt{n \log n})}$.*

Proof. Let p_i denote the i-th prime number. For any integer $n \geqslant 2$, let $P(n) = p_k$ be the largest prime number such that $p_1 + p_2 + \cdots + p_k \leqslant n$. We define

$$L_n = (\mathtt{a}^{p_1})^+\mathtt{b}(\mathtt{a}^{p_2})^+\mathtt{b}\cdots(\mathtt{a}^{p_k})^+\mathtt{b}.$$

Then L_n is accepted by a DFA with $O(n)$ states.

If k is itself prime, the shortest power in L_n is $w = (\mathtt{a}^\ell\mathtt{b})^k$, where $\ell = p_1p_2\cdots p_k$. For $n \geqslant 2$, let

$$F(n) = \prod_{p \leqslant P(n)} p,$$

where the product is over primes p. We have $F(n) \in e^{\Omega(\sqrt{n \log n})}$ [24, Theorem 1]. This lower bound is valid for all sufficiently large n; in particular, it holds for infinitely many n such that $n = p_1 + p_2 + \cdots + p_k$, where k is prime. This gives the desired result. □

9. Additional results on powers

Dömösi et al. [8, Theorem 10] proved that if L is a slender regular language over Σ, and Q_Σ is the set of primitive words over Σ, then $L \cap Q_\Sigma$ is regular. This result is somewhat surprising, since it is widely believed that Q_Σ is not even context-free for $|\Sigma| \geqslant 2$. In this section, we apply a variation of their argument to show that Q_Σ may be replaced by the language of squares, (cubes, etc.) over Σ.

For any integer $k \geqslant 2$ and alphabet Σ, let $P(k, \Sigma)$ denote the set of k-powers over Σ. Clearly, for $|\Sigma| \geqslant 2$, $P(k, \Sigma)$ is not context-free.

Proposition 48. *If $L \subseteq \Sigma^*$ is a slender regular language, then for all integers $k \geqslant 2$, $L \cap P(k, \Sigma)$ is regular.*

Proof. If L is slender, then by Theorem 11 it suffices to consider $L = uv^*w$. The result is clearly true if v is empty, so we suppose v is non-empty. Let x and y be the primitive roots of v and wu, respectively. If $x = y$, then the set of k-powers in v^*wu is given by $v^*wu \cap (x^k)^*$, so the set of k-powers in uv^*w is regular. If $x \neq y$, then by Theorem 29, the set v^*wu contains only finitely many k-powers. The set of k-powers in uv^*w is therefore finite, and, a fortiori, regular. □

10. Testing if an NFA accepts a bordered word

In this section, we give an efficient algorithm to test if an NFA accepts a bordered word. We also give upper and lower bounds on the length of a shortest bordered word accepted by an NFA.

Proposition 49. *Given an NFA M with n states and t transitions, we can decide if M accepts at least one bordered word in $O(n^3t^2)$ time.*

Proof. Given an NFA $M = (Q, \Sigma, \delta, q_0, F)$, we can easily create an NFA-ϵ M' that accepts

$$\{u \in \Sigma^* \ : \ \text{there exists } w \in \Sigma^* \text{ such that } uwu \in L\}$$

by "guessing" the state we would be in after reading uw, and then verifying it. More formally, we let $M' = (Q', \Sigma, \delta', q_0', F')$ where

- $Q' = \{q_0'\} \cup \{[p, q, r] \ : \ p, q, r \in Q\}$ and
- $F' = \{[p, q, r] \ : \ r \in F$ and there exists $w \in \Sigma^*$ such that $q \in \delta(p, w)\}$.

The transitions are defined as follows: $\delta(q_0', \epsilon) = \{[q_0, p, p] \ : \ p \in Q\}$ and

$$\delta([p, q, r], a) = \Big\{[p', q, r'] \ : \ p' \in \delta(p, a), r' \in \delta(r, a)\Big\}.$$

If M has n states and t transitions, then M' has $n^3 + 1$ states and at most $n + n^3t^2$ transitions. Now get rid of all useless states and their associated transitions. We can compute the final states by doing n depth-first searches, starting at each node, at a cost of $O(n(n + t))$ time. Now we just test to see if $L(M')$ accepts a non-empty string, which can be done in linear time in the size of M'. □

Corollary 50. *If M is an NFA with n states, and it accepts at least one bordered word, it must accept a bordered word of length $< 2n^2 + n$.*

Proof. Consider the DFA M' constructed in the proof of the previous theorem, which accepts

$$L' = \{u \in \Sigma^* \ : \ \text{there exists } w \in \Sigma^* \text{ such that } uwu \in L\}.$$

If M accepts a bordered string, then M' accepts a non-empty string. Although M' has $n^3 + 1$ states, once a computation leaves q_0' and enters a triple of the form $[p, q, r]$, it never enters a state $[p', q', r']$ with $q \neq q'$. Thus we may view the NFA M' as implicitly defining a union of n disjoint languages, each accepted by an NFA with n^2 states. Therefore, if M' accepts a non-empty string u, it accepts one of length at most n^2. Now the corresponding bordered string is uwu. The string w is implicitly defined in the previous proof as a path from a state p to a state q. If such a path exists, it is of length at most $n - 1$. Thus there exists $uwu \in L(M)$ with $|uwu| \leqslant 2n^2 + n - 1$. □

Proposition 51. *For infinitely many n there is an DFA of n states such that the shortest bordered word accepted is of length $n^2/2 - 6n + 43/2$.*

Proof. Consider $a(b^t)^+ca(b^{t-1})^+c$. An obvious DFA can accept this using $2t + 5$ states. However, the shortest bordered word accepted is $ab^{t(t-1)}cab^{t(t-1)}c$, which is of length $2t(t - 1) + 4 = n^2/2 - 6n + 43/2$. □

We now consider testing if an NFA accepts infinitely many bordered words.

Corollary 52. *If an NFA M has n states and t transitions, we can test whether M accepts infinitely many bordered words in $O(n^6t^2)$ time.*

Proof. If an NFA M accepts infinitely many words of the form uwu, there are two possibilities, at least one of which must hold:

(a) there is a single word u such that there are infinitely many w with $uwu \in L(M)$ or
(b) there are infinitely many u, with possibly different w depending on u, such that $uwu \in L(M)$.

To check these possibilities, we return to the NFA-ϵ M' constructed in the proof of Theorem 49. First, for each pair of states q_i to q_j, we determine whether there exists a non-empty path from q_i to q_j. This can be done with n different depth-first searches, starting at each vertex, at a cost of $O(n^3(n^3+t^2))$ time. In particular, for each vertex, we learn whether there is a non-empty cycle beginning and ending at that vertex.

Now let us check whether (a) holds. After removing all useless states and their associated transitions, look at the remaining final states $[p, q, r]$ of M' and determine if there is a path from p to q that goes through a vertex with a cycle. This can be done by testing, for each vertex s that has a cycle, whether there is a non-empty path from p to s and then s to q. If such a vertex exists, then there are infinitely many w in some uwu.

To check whether (b) holds, we just need to know whether M' accepts infinitely many strings, which we can easily check by looking for a directed cycle.

The total cost is therefore $O(n^3(n^3t^2))$. □

We now prove the following decomposition theorem for regular languages consisting only of bordered words.

Theorem 53. *If every word in a regular language L is bordered, then there is a decomposition of L as a finite union of regular languages of the form JKJ, where each J and K are regular and $\epsilon \notin J$.*

Proof. Let L be accepted by an NFA $M = (Q, \Sigma, \delta, q_0, F)$. For each $x \in \Sigma^+$, define an automaton $M_x = (Q, \Sigma, \delta, I', F')$ (for M_x we permit multiple initial states), where the set of initial states is $I' = \delta(q_0, x)$, and the set of final states is $F' = \{q \in Q : \delta(q, x) \in F\}$. Then M_x has the property that for every $w \in L(M_x)$, we have $xwx \in L(M)$. Note that there are only finitely many distinct automata M_x.

For each automaton M_x, define the regular language

$$L_x = \Big\{ y : \delta(q, y) = I' \quad \text{and} \quad \{q \in Q : \delta(q, y) \in F\} = F' \Big\}.$$

Note that again there are only finitely many distinct languages L_x.

For every $x \in \Sigma^+$, every word in $L_xL(M_x)L_x$ is in L. Furthermore, if $w \in L$ is bordered, then there exists $x \in \Sigma^+$ such that $w \in L_xL(M_x)L_x$. Thus, if every word of L is bordered, then $L = \cup_{x\in\Sigma^+} L_xL(M_x)L_x$. Since there are only finitely many languages L_x and $L(M_x)$, this union is finite, as required. □

11. Testing if an NFA accepts an unbordered word

We present a simple test to determine if all words in a regular language are bordered, and to determine if a regular language contains infinitely many unbordered words. We first need the following well-known result about words, which is due to Lyndon and Schützenberger [21].

Lemma 54. *Suppose x, y and z are non-empty words, and that $xy = yz$. Then there is a non-empty word p, a word q and a non-negative integer k_1 for which we can write $x = pq, z = qp$, and $y = (pq)^{k_1}p$.*

We also need the following result, which is just a variation of the pumping lemma.

Lemma 55. *Let $M = (Q, \Sigma, \delta, q_0, F)$ be an n-state NFA. Let L be the language accepted by M. Let d be a positive integer. Let (X, y, Z) be a 3-tuple of words for which $|y|$ is a multiple of d, $|y| \geq nd$ and $XyZ \in L$. Then there are words r, s and t, whose lengths are multiples of d, with $|s| \geq d$, for which we can write $y = rst$, and, for all $z \geq 0$, $Xrs^ztY \in L$.*

Proof. Set $l := |X|$ and $m := |y|/d$, $\gamma := XyZ$, and $k := |\gamma|$. First, write γ as a sequence of letters, that is, $\gamma := \gamma_1\gamma_2\cdots\gamma_k$ with each γ_i a letter. By $\gamma[i, j]$ for $1 \leq i, j \leq |\gamma|$ we mean the subsequence that consists of the $i - j + 1$ consecutive letters of γ starting at position i and ending at position j, that is, $\gamma_i\gamma_{i+1}\cdots\gamma_j$. If $i > j$ we take $\gamma[i, j]$ to be the empty word. Now we have the following sequence of k states

$$q_1 \in \delta(q_0, \gamma_1), q_2 \in \delta(q_1, \gamma_2), \ldots, q_k \in \delta(q_{k-1}, \gamma_k).$$

We will choose q_k to be a final state.

Note that $y = \gamma[l+1, l+md]$, and consider the following sequence of $m + 1$ states of M:

$$q_l, q_{l+d}, q_{l+2d}, \ldots, q_{l+md}.$$

There are integers i and j, with $0 \leq i < j \leq m$ for which $q_{l+id} = q_{l+jd}$. Set $r := \gamma[l+1, l+id], s := \gamma[l+id+1, l+jd]$, and $t := \gamma[l+jd+1, l+md]$, so $y = rst$. Note that $|s| \geq d$, and the desired conclusion follows immediately. □

Lemma 56. *Let M be an n-state NFA. Let L be the language accepted by M. Let (X, Y, Z) be a 3-tuple of words for which $XYZ \in L$. Then there is a word y for which $|y| < n$ and $XyZ \in L$.*

Proof. Let $S := \{u \in \Sigma^* : XuZ \in L\}$. Let y be an element of S of minimal length. We proceed by contradiction, and suppose $|y| \geq n$. We apply Lemma 55 to (X, y, Z), with $d = 1$, and write $y = rst$ with s non-empty. Then $XrtZ \in L$, which violates the minimality of $|y|$. □

Lemma 57. *Suppose there are words Ψ_L, Ψ_R, e, f, g and h with $|\Psi_L| = |\Psi_R|$, $|e| < |\Psi_L|$, $|g| < |\Psi_L|$, and for which*

$$b_\zeta := \Psi_L e = f\Psi_R \tag{1}$$

and

$$b_\eta := \Psi_L g = h\Psi_R. \tag{2}$$

Suppose further that $|b_\eta| < |b_\zeta|$. Then we can write $\Psi_L = h(pq)^k p$ and $\Psi_R = (pq)^k pg$ for p a non-empty word, q a word for which $|g| + |pq| = |f|$, and k a positive integer.

Proof. Since $|b_\eta| < |b_\zeta|$, we must have $|g| < |e| < |\Psi_R|$. This last observation, together with (1) and (2) above allows us to assert that there are non-empty words s_1 and s_2, with $|s_2| > |s_1|$, such that $\Psi_R = s_1 e = s_2 g$. This last fact combined again with (1) and (2) yields that

$$\Psi_L = fs_1 = hs_2 \tag{3}$$

and

$$\Psi_R = s_1 e = s_2 g. \tag{4}$$

Now we can apply (3) and (4) to assert that there are non-empty words r_1 and r_2 for which $s_1 r_1 = s_2 = r_2 s_1$; that is,

$$s_1 r_1 = r_2 s_1. \tag{5}$$

Now apply Lemma 54 to (5) to get that there is a non-empty word p, a word q and an integer $k_1 \geq 0$ for which $s_1 = (pq)^{k_1} p$, $r_1 = qp$, and $r_2 = pq$. Set $k := k_1 + 1$. Then $s_2 = (pq)^k p$, and (3) gives $\Psi_L = h(pq)^k p$, and (4) gives $\Psi_R = (pq)^k pg$. Also $s_2 = r_2 s_1$ combined with (3) above gives that $f = hr_2$, so $|g| + |pq| = |h| + |pq| = |h| + |r_2| = |f|$. □

Theorems 58 and 67 below are the main results.

Theorem 58. *Let M be an n-state NFA. Let L be the language accepted by M. Let N be a non-negative integer. Suppose all words in L of length in the interval $[N, 2N + 6n + 1]$ are bordered. Then all words in L of length greater than $2N + 6n + 1$ are bordered. Hence, if all words in L of length at most $6n + 1$ are bordered, then all the words in L must be bordered.*

Proof. We will prove Theorem 58 by making the following series of observations. Throughout, we will assume that all words in L of length in the interval $[N, 2N + 6n + 1]$ are bordered, and we will assume w is an unbordered word in L for which $|w| > 2N + 6n + 1$, with $|w|$ minimal. We write w as $u\theta v$ with θ a word for which $|\theta| \leq 1$ and u and v words for which $|u| = |v| > 3n + N$.

Claim 59. *Write u as $\Psi_L X_L$ and v as $X_R \Psi_R$, for words Ψ_L, X_L, Ψ_R, X_R for which $|X_L| = |X_R| = n$. (So that w is $\Psi_L X_L \theta X_R \Psi_R$.) Then there are words x_L and x_R, both of length less than n, for which:*

(i) $\zeta := \Psi_L x_L \theta X_R \Psi_R \in L$ *and*
(ii) $\eta := \Psi_L X_L \theta x_R \Psi_R \in L$.

Further, $N \leq |\zeta| < |w|$, and $N \leq |\eta| < |w|$.

To justify (i), apply Lemma 56 to the 3-tuple $(\Psi_L, X_L, \theta X_R \Psi_R)$. Similarly, to arrive at (ii), apply Lemma 56 again to the 3-tuple $(\Psi_L X_L \theta, X_R, \Psi_R)$.

Claim 60. *We can write $\Psi_L = h(pq)^k p$ and $\Psi_R = (pq)^k pg$ for p a non-empty word, g, h and q words for which $|g| = |h|$, $|pq| + |g| \leq n$, and k a positive integer. Hence w can be written as $h(pq)^k p X_L \theta X_R (pq)^k pg$.*

To justify Claim 60, first recall $w = \Psi_L X_L \theta X_R \Psi_R$ and $|\Psi_L| = |\Psi_R| > 2n$. From Claim 59 above we get that ζ and η are bordered words, so we can assert that there exist non-empty words b_ζ and b_η, and words p_ζ and p_η, for which:

(I) $\zeta = \Psi_L x_L \theta X_R \Psi_R = b_\zeta p_\zeta b_\zeta$ and
(II) $\eta = \Psi_L X_L \theta x_R \Psi_R = b_\eta p_\eta b_\eta$.

Note that, if $|b_\zeta| \leq |\Psi_L|$ then by (I) b_ζ would be a border for w. So we must have $|b_\zeta| > |\Psi_L|$. Similarly, (II) gives that $|b_\eta| > |\Psi_L|$. These latter facts together with (I) and (II) give that there exists non-empty words e, f, g, h, for which $|e| = |f|$, $|g| = |h|$, and for which

$$b_\zeta = \Psi_L e = f \Psi_R \tag{6}$$

and

$$b_\eta = \Psi_L g = h \Psi_R. \tag{7}$$

Further, $|\zeta| < |w|$ implies that $|f| \leq n$, and similarly $|\eta| < |w|$ implies that $|h| \leq n$.

Suppose $|b_\eta| = |b_\zeta|$. Then from (6) and (7) above, $|e| = |g|$. But e and g are suffixes of Ψ_R, so we get that $e = g$. Hence $b_\zeta = \Psi_L e = \Psi_L g = b_\eta$. Set $b := b_\zeta = b_\eta$. Then from (II) above, as $|b| \leq |\Psi_L| + n$, b is a prefix of $\Psi_L X_L$. And from (I) above, b is a suffix of $X_R \Psi_R$. So b is a non-empty prefix of w, and a suffix of w. Hence, as $|b| \leq \frac{|w|}{2}$, b is a border for w.

So we must have $|b_\eta| \neq |b_\zeta|$. Suppose first that $|b_\eta| < |b_\zeta|$. Now apply Lemma 57 to get that there is a positive integer k, a non-empty word p and a word q for which $\Psi_L = h(pq)^k p$ and $\Psi_R = (pq)^k pg$. And finally observe that $|pq| + |g| = |f| \leq n$. If $|b_\eta| > |b_\zeta|$, the argument is similar, so Claim 60 is established.

Claim 61. *Let $x := pq$ in the statement of Claim 60. There is a conjugate c_L of x which is a prefix of Ψ_L, and there is a conjugate c_R of x which is a suffix of Ψ_R.*

To justify Claim 61, let S_L be the prefix of length n of Ψ_L. So there is a word T_L for which we can write $\Psi_L X_L \theta X_R = S_L T_L$. (So w is $S_L T_L \Psi_R$.) Now apply Lemma 56 to (S_L, T_L, Ψ_R), obtaining a word t_L, with $|t_L| < n$ for which $w_1 := S_L t_L \Psi_R \in L$. By supposition, since $N \leq |w_1| < |w|$, w_1 has a border, say b_1. Further, if $|b_1| \leq n$ then b_1 would be a border for w. So we must have $|b_1| > n$. And $|b_1| \leq \frac{|w_1|}{2}$ implies $|b_1| \leq |\Psi_R|$.

So b_1 is a suffix of Ψ_R of length greater than n; hence by Claim 60 above we can write $b_1 = s_x x^{k_2} pg$ for some integer $k_2 \geq 0$, with s_x a suffix of x. Write $x = p_x s_x$, and recall that p is a prefix of x. Then $|s_x x^{k_2} pg| > n$ and $|x| + |g| \leq n$ (from Claim 60) yields that $s_x p_x$ is a prefix of $s_x x^{k_2} pg$, that is, $s_x p_x$ is a prefix of b_1. So set $c_L := s_x p_x$. Since b_1 is a prefix of w_1, c_L must be a prefix of w_1, and $|c_L| \leq n = |S_L|$ gives that c_L is a prefix of S_L, and the first statement of Claim 61 follows.

To get the second statement of Claim 61, similarly let S_R be the suffix of length n of Ψ_R. So there is a word T_R for which we can write $X_L \theta X_R \Psi_R = T_R S_R$. (So w is $\Psi_L T_R S_R$.) Now apply Lemma 56 to (Ψ_L, T_R, S_R), obtaining a word t_R, with $|t_R| < n$ for which $w_2 := \Psi_L t_R S_R \in L$. By supposition, since $N \leq |w_2| < |w|$, w_2 has a border, say b_2. Further, if $|b_2| \leq n$ then b_2 would be a border for w. So we can assert that $n < |b_2| \leq |\Psi_L|$.

So b_2 is a prefix of Ψ_L of length greater than n; hence by Claim 60 we can write $b_2 = hx^{k_3}\rho_x$ for some integer $k_3 \geq 0$, with ρ_x a prefix of x. Write $x = \rho_x \sigma_x$. Then $|hx^{k_3}\rho_x| > n$ and $|x| + |h| \leq n$ (from Claim 60) yields that $\sigma_x \rho_x$ is a suffix of $hx^{k_3}\rho_x$, that is, $\sigma_x \rho_x$ is a suffix of b_2. So set $c_R := \sigma_x \rho_x$. Since b_2 is a suffix of w_2, c_R must be a suffix of w_2, and also $|c_R| \leq n = |S_R|$ yields that c_R is a suffix of S_R, and the second statement of Claim 61 follows.

To complete the proof of Theorem 58, note that, since c_L and c_R are both conjugates of x, c_L and c_R are non-empty words which are conjugates. So there is a non-empty word α and a word β for which we can write $c_L = \alpha\beta$ and $c_R = \beta\alpha$. Then α is a prefix of Ψ_L, and α is a suffix of Ψ_R, which gives that α is a border for w, and gives a contradiction. □

Corollary 62. *The problem of determining if an NFA accepts an unbordered word is decidable.*

Proof. Let M be an NFA with n states. To determine if M accepts an unbordered word, it suffices to test whether M accepts an unbordered word of length at most $6n + 1$. □

We do not know if there is a polynomial-time algorithm to test if an NFA accepts an unbordered word or if the problem is computationally intractable.

Theorem 58 gives an upper bound of $6n + 1$ on the length of a shortest unbordered word accepted by an n-state NFA. The best lower bound we are able to come up with is $2n - 3$, as illustrated by the following example: an NFA of n states accepts $ab^{n-3}ab^*$, and the shortest unbordered word accepted is $ab^{n-3}ab^{n-2}$, which is of length $2n - 3$ (Fig. 3).

Theorem 63. *Let M be an n-state NFA, and let L be the language accepted by M. Suppose there is an unbordered word in L of length greater than $4n^2 + 6n + 1$. Then L contains infinitely many unbordered words.*

Proof. Suppose L contains only finitely many unbordered words. Let w be an unbordered word in L of length greater than $4n^2 + 6n + 1$, with $|w|$ maximal. Write w as $\Psi_L X_L \theta X_R \Psi_R$ for words $\Psi_L, X_L, \theta, \Psi_R, X_R$ for which $|X_L| = |X_R| = n$, $|\Psi_L| = |\Psi_R| > 2n^2 + 2n$, and $|\theta| \leq 1$. We proceed by making the following series of observations.

L	decide if $L(M) \cap L = \emptyset$	decide if $L(M) \cap L$ infinite	upper bound on shortest element of $L(M) \cap L$	worst-case lower bound known
palindromes	$O(n^2 + t^2)$	$O(n^2 + t^2)$	$2n^2 - 1$	$n^2/2 - 3n + 5$
non-palindromes	$O(n^2 + tn)$	$O(n^2 + t^2)$	$3n - 1$	$3n - 1$
k-powers (k fixed)	$O(n^{2k-1}t^k)$	$O(n^{2k-1}t^k)$	kn^k	$\Omega(n^k)$
k-powers (k part of input)	PSPACE-complete	PSPACE-complete		
non-k-powers	$O(n^3 + tn^2)$	$O(n^3 + tn^2)$	$3n$	$(2 + \frac{1}{2k-2})n - O(1)$
powers	PSPACE-complete	PSPACE-complete	$(n+1)n^{n+1}$	$e^{\Omega(\sqrt{n \log n})}$
non-powers	$O(n^5)$	$O(n^5)$	$3n$	$(5/2)n - 2$
bordered words	$O(n^3t^2)$	$O(n^6t^2)$	$2n^2 + n - 1$	$n^2/2 - 6n + 43/2$
unbordered words	decidable	decidable	$6n + 1$	$2n - 3$

Fig. 3. Summary of main results.

Claim 64. *There are words x_L, u_L, y_L and x_R, u_R, y_R, with u_L and u_R both non-empty, $X_L = x_L u_L y_L, X_R = x_R u_R y_R$, and for which:*

(i) $\zeta := \Psi_L x_L u_L u_L y_L \theta X_R \Psi_R \in L$ *and*
(ii) $\eta := \Psi_L X_L \theta x_R u_R u_R y_R \Psi_R \in L$.

Further, $|\zeta| > |w|$ *and* $|\eta| > |w|$.

To justify (i), apply Lemma 55 (with $d = 1$) to the 3-tuple $(\Psi_L, X_L, \theta X_R \Psi_R)$. Similarly, to arrive at (ii), apply Lemma 55 again (also with $d = 1$) to the 3-tuple $(\Psi_L X_L \theta, X_R, \Psi_R)$.

Claim 65. *We can write $\Psi_L = h(pq)^k p$ and $\Psi_R = (pq)^k pg$ for p a non-empty word, g, h and q words for which $|g| = |h|$, $|pq| + |g| \leq 2n$, and k an integer $\geq n$. Hence w can be written as $h(pq)^k p X_L \theta X_R (pq)^k pg$.*

To justify Claim 65, first recall that $w = \Psi_L x_L u_L y_L \theta x_R u_R y_R \Psi_R$, and $X_L = x_L u_L y_L$, $X_R = x_R u_R y_R$. From Claim 64 above and the maximality of $|w|$ we get that ζ and η are bordered words, so we can assert that there exist non-empty words b_ζ and b_η, and words p_ζ and p_η, for which:

(I) $\zeta = \Psi_L x_L u_L u_L y_L \theta X_R \Psi_R = b_\zeta p_\zeta b_\zeta$ and
(II) $\eta = \Psi_L X_L \theta x_R u_R u_R y_R \Psi_R = b_\eta p_\eta b_\eta$.

Note that, if $|b_\zeta| \leq |\Psi_L|$ then by (I) b_ζ would be a border for w. So we must have $|b_\zeta| > |\Psi_L|$. Similarly, (II) gives that $|b_\eta| > |\Psi_L|$. These latter facts together with (I) and (II) give that there exists non-empty words e, f, g, h, for which $|e| = |f|$, $|g| = |h|$, and for which

$$b_\zeta = \Psi_L e = f \Psi_R \tag{8}$$

and

$$b_\eta = \Psi_L g = h \Psi_R. \tag{9}$$

Further, the reader can verify that $|e| \leq 2n < |\Psi_R|$, and $|g| \leq 2n < |\Psi_R|$.

Suppose $|b_\eta| = |b_\zeta|$. Then from (8) and (9) above, $|e| = |g|$. But e and g are suffixes of Ψ_R, so we get that $e = g$. Hence $b_\zeta = \Psi_L e = \Psi_L g = b_\eta$. Set $b := b_\zeta = b_\eta$. Now $|u_L y_L \theta X_R| > |x_L u_L|$, so from (I) above, we must have $|b| \leq |u_L y_L \theta X_R \Psi_R|$, that is, b is a suffix of $u_L y_L \theta X_R \Psi_R$. Similarly, $|X_L \theta x_R u_R| > |u_R y_R|$, so from (II) above we get that $|b| \leq |\Psi_L X_L \theta x_R u_R|$, that is, b is a prefix of $\Psi_L X_L \theta x_R u_R$. So b is a non-empty prefix of w, and a suffix of w. Hence w must be bordered, which is a contradiction.

So we must have $|b_\eta| \neq |b_\zeta|$. First, suppose $|b_\eta| < |b_\zeta|$. Now apply Lemma 57 to get that there is a positive integer k, a non-empty word p and a word q for which $\Psi_L = h(pq)^k p$ and $\Psi_R = (pq)^k pg$. And finally observe that $|pq| + |g| = |f| \leq 2n$, and since $|\Psi_L| > 2n^2 + 2n$ and $|pq| \leq 2n$, we get that $k \geq n$. The case $|b_\eta| > |b_\zeta|$ is symmetric, so Claim 65 is established.

Claim 66. *Let $x := pq$ in the statement of Claim 65. There is a conjugate c_L of x which is a prefix of Ψ_L, and there is a conjugate c_R of x which is a suffix of Ψ_R.*

To justify Claim 66, recall from Claim 65 that w is $\Psi_L X_L \theta X_R x^k pg$. And since $k \geq n$, we can apply Lemma 55 to the 3-tuple of words $(\Psi_L X_L \theta X_R, x^k, pg)$, with $d := |x|$, obtaining a positive integer J_1 for which, for all $z \geq 0$, we have $\Psi_L X_L \theta X_R x^{k+J_1 z} pg \in L$. So choose $z_1 := |\Psi_L X_L \theta X_R|$, and define $w_1 := \Psi_L X_L \theta X_R x^{k+J_1 z_1} pg$. By supposition w_1 is a bordered word, say with border b_1. Further, if $|b_1| \leq |\Psi_R|$ then b_1 would be a border for w. So we must have $|b_1| > |\Psi_R|$. And $|b_1| \leq \frac{|w_1|}{2}$ implies $|b_1| \leq |x^{k+J_1 z_1} pg|$.

So b_1 is a suffix of $x^{k+J_1 z_1} pg$ of length greater than $|\Psi_R| > 2n$, hence by Claim 65 above we can write $b_1 = s_x x^{k_2} pg$ for some integer $k_2 \geq 0$, with s_x a suffix of x. Write $x = p_x s_x$, and recall that p is a prefix of x. Then $|s_x x^{k_2} pg| > 2n$ and $|x| + |g| \leq 2n$ (from Claim 65) yields that $s_x p_x$ is a prefix of $s_x x^{k_2} pg$, that is, $s_x p_x$ is a prefix of b_1. So set $c_L := s_x p_x$. Since b_1 is a prefix of w_1, c_L must be a prefix of w_1, and $|c_L| \leq 2n$ gives that c_L is a prefix of Ψ_L, and the first statement of Claim 66 follows.

To justify the second statement of Claim 66, we proceed similarly; that is, we recall that w is $hx^k pX_L \theta X_R \Psi_R$, and apply Lemma 55 to the 3-tuple of words $(h, x^k, pX_L \theta X_R \Psi_R)$, with $d := |x|$, allowing us to assert that there is a positive integer J_2 for which, for all $z \geq 0$, we have $hx^{k+J_2 z} pX_L \theta X_R \Psi_R \in L$. So choose $z_2 := |pX_L \theta X_R \Psi_R|$, and define $w_2 := hx^{k+J_2 z_2} pX_L \theta X_R \Psi_R$. By supposition w_2 is a bordered word, say with border b_2. Further, if $|b_2| \leq |\Psi_L|$ then b_2 would be a border for w. So we must have $|b_2| > |\Psi_L|$. And $|b_2| \leq \frac{|w_2|}{2}$ implies $|b_2| \leq |hx^{k+J_2 z_2} p|$.

So b_2 is a prefix of $hx^{k+J_2 z_2} p$ of length greater than $|\Psi_L| > 2n$; hence by Claim 65 we can write $b_2 = hx^{k_3} \rho_x$ for some integer $k_3 \geq 0$, with ρ_x a prefix of x. Write $x = \rho_x \sigma_x$. Then $|hx^{k_3} \rho_x| > 2n$ and $|x| + |h| \leq 2n$ (from Claim 65) yields that $\sigma_x \rho_x$ is a suffix of $hx^{k_3} \rho_x$, that is, $\sigma_x \rho_x$ is a suffix of b_2. So set $c_R := \sigma_x \rho_x$. Since b_2 is a suffix of w_2, c_R must be a suffix of w_2, and also $|c_R| \leq 2n$ yields that c_R is a suffix of Ψ_R, and the second statement of Claim 66 follows.

To complete the proof of Theorem 63, note that, since c_L and c_R are both conjugates of x, c_L and c_R are non-empty words which are conjugates. So there is a non-empty word α and a word β for which we can write $c_L = \alpha\beta$ and $c_R = \beta\alpha$. Then α is a prefix of Ψ_L, and α is a suffix of Ψ_R, which gives that α is a border for w, which is a contradiction. So we are forced to conclude that L contains infinitely many unbordered words. □

Theorem 67. *Let M be an n-state NFA, and let L be the language accepted by M. Then the following are equivalent:*

1. *L contains infinitely many unbordered words.*
2. *There is an unbordered word w in L, with $4n^2 + 6n + 2 \leq |w| \leq 8n^2 + 18n + 5$.*

Proof. (1) → (2). Suppose all words $w \in L$ whose lengths are in $[4n^2 + 6n + 2, 8n^2 + 18n + 5]$ are bordered words. Then by Theorem 58 (with $N = 4n^2 + 6n + 2$) we have that any word in L whose length is at least $4n^2 + 6n + 2$ is bordered, i.e., L contains at most finitely many unbordered words.

(2) → (1). This follows immediately from Theorem 63. □

Corollary 68. *The problem of determining if an NFA accepts infinitely many unbordered words is decidable.*

Proof. Let M be an NFA with n states. To determine if M accepts infinitely many unbordered words, it suffices to test whether M accepts an unbordered word w, where $4n^2 + 6n + 2 \leq |w| \leq 8n^2 + 18n + 5$. □

We do not know if there is a polynomial-time algorithm to test if an NFA accepts infinitely many unbordered words or if the problem is computationally intractable.

12. Final remarks

In this paper, we examined the complexity of checking various properties of regular languages, such as consisting only of palindromes, containing at least one palindrome, consisting only of powers, or containing at least one power. In each case (except for the unbordered words), we were able to provide an efficient algorithm or show that the problem is likely to be hard. Our results are summarized in Fig. 3. Here M is an NFA with n states and t transitions. When L is the language of unbordered words, it is an open problem to either find polynomial-time algorithms to test if (a) $L(M) \cap L = \emptyset$, and (b) $L(M) \cap L$ is infinite, or to show the intractability of these problems.

Acknowledgments

The algorithm mentioned in Section 2 for testing if an NFA-ϵ accepts infinitely many words was suggested to us by Timothy Chan. We would like to thank both him and Jack Zhao for their ideas on this subject.

References

[1] M. Ackerman, J. Shallit, Efficient enumeration of regular languages, in: Proceedings of the Implementation and Application of Automata, 12th International Conference (CIAA 2007), vol. 4783, Lecture Notes in Computer Science, Springer-Verlag, 2007, pp. 226–242.

[2] A. Aho, J. Hopcroft, J. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, 1974.

[3] J.-C. Birget, Intersection and union of regular languages and state complexity, Inform. Process. Lett. 43 (1992) 185–190.

[4] G. Castiglione, A. Restivo, S. Salemi, Patterns in words and languages, Discrete Appl. Math. 144 (2004) 237–246.

[5] M. Chrobak, Finite automata and unary languages, Theor. Comput. Sci. 47 (1986) 149–158. (Errata 302 (2003) 497–498).

[6] T. Cormen, C. Leiserson, R. Rivest, C. Stein, Introduction to Algorithms, second ed., MIT Press, 2001.

[7] P. Dömösi, G. Horváth, M. Ito, A small hierarchy of languages consisting of non-primitive words, Publ. Math. (Debrecen) 64 (2004) 261–267.

[8] P. Dömösi, C. Martín-Vide, V. Mitrana, Remarks on sublanguages consisting of primitive words of slender regular and context-free languages, in: J. Karhumäki et al. (Eds.), Theory is Forever, vol. 3113, Lecture Notes in Computer Science, Springer-Verlag, 2004, pp. 60–67.

[9] K. Ellul, B. Krawetz, J. Shallit, M. Wang, Regular expressions: new results and open problems, J. Automata Lang. Combin. 9 (2004) 233–256.

[10] M. Garey, D. Johnson, Computers and Intractability, Freeman, 1979.

[11] I. Glaister, J. Shallit, A lower bound technique for the size of nondeterministic finite automata, Inform. Process. Lett. 59 (1996) 75–77.

[12] G.H. Hardy, E.M. Wright, An Introduction to the Theory of Numbers, fifth ed., Oxford, 1979.

[13] D.R. Heath-Brown, Zero-free regions for Dirichlet L-functions, and the least prime in an arithmetic progression, Proc. Lond. Math. Soc. 64 (1992) 265–338.

[14] J.E. Hopcroft, J.D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, 1979.

[15] S. Horváth, J. Karhumäki, J. Kleijn, Results concerning palindromicity, J. Inf. Process. Cybern. EIK 23 (1987) 441–451.

[16] H.B. Hunt III, On the time and tape complexity of languages. I, in: Proceedings of the Fifth Annual ACM Symposium Theoretical Computation ACM, 1973, pp. 10–19.

[17] M. Ito, M. Katsura, H.J. Shyr, S.S. Yu, Automata accepting primitive words, Semigroup Forum 37 (1988) 45–52.

[18] J.-Y. Kao, J. Shallit, Z. Xu, The noncommutative Frobenius problem, in preparation.

[19] D. Knuth, J. Morris Jr., V. Pratt, Fast pattern matching in strings, SIAM J. Comput. 6 (1977) 323–350.

[20] M. Kunze, H.J. Shyr, G. Thierrin, h-Bounded and semi-discrete languages, Inform. Control 51 (1981) 147–187.

[21] R.C. Lyndon, M.-P. Schützenberger, The equation $a^m = b^n c^p$ in a free group, Mich. Math. J. 9 (1962) 289–298.

[22] E. Mäkinen, On lexicographic enumeration of regular and context-free languages, Acta Cybern. 13 (1997) 55–61.

[23] A. Martinez, Efficient computation of regular expressions from unary NFAs, in: Proceedings of the Descriptional Complexity of Formal Systems (DCFS 2002), 2002, pp. 174–187.

[24] W. Miller, The maximum order of an element of a finite symmetric group, Am. Math. Mon. 94 (1987) 497–506.

[25] P. Pritchard, Linear prime-number sieves: a family tree, Sci. Comput. Program. 9 (1987) 17–35.

[26] G. Păun, A. Salomaa, Thin and slender languages, Discrete Appl. Math. 61 (1995) 257–270.

[27] A. Restivo, S. Salemi, Words and patterns, in: Proceedings of the Developments in Language Theory, 5th International Conference (DLT 2001), vol. 2295, Lecture Notes in Computer Science, Springer-Verlag, 2002, pp. 117–129.

[28] L. Rosaz, Puzzle corner, #50, Bull. Eur. Assoc. Theor. Comput. Sci. (76) (2002) 234, solution in No. 77 (June 2002) 261.

[29] W. Savitch, Relationships between nondeterministic and deterministic tape complexities, J. Comput. Syst. Sci. 4 (1970) 177–192.

[30] J. Shallit, Numeration systems, linear recurrences, and regular sets, Inform. Comput. 113 (1994) 331–347.

[31] J. Shallit, Y. Breitbart, Automaticity I: properties of a measure of descriptional complexity, J. Comput. Syst. Sci. 53 (1996) 10–25.

[32] H.J. Shyr, S.S. Yu, Non-primitive words in the language p^+q^+, Soochow J. Math. 20 (1994) 535–546.

[33] S. Yu, Regular languages, in: G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages, Springer-Verlag, 1997, pp. 41–110. (Chapter 1).

[34] G.-Q. Zhang, Automata, boolean matrices, and ultimate periodicity, Inform. Comput. 152 (1999) 138–154.