On the Gap Between Separating Words and Separating Their Reversals

Farzam Ebrahimnejad[a]

[a] *Department of Computer Engineering, Sharif University of Technology, Tehran, Iran*

Abstract

A deterministic finite automaton (DFA) separates two strings w and x if it accepts w and rejects x. The minimum number of states required for a DFA to separate w and x is denoted by $\operatorname{sep}(w, x)$. The present paper shows that the difference $\left|\operatorname{sep}(w, x) - \operatorname{sep}(w^R, x^R)\right|$ is unbounded for a binary alphabet; here w^R stands for the mirror image of w. This solves an open problem stated in [Demaine, Eisenstat, Shallit, Wilson: Remarks on separating words. DCFS 2011. LNCS vol. 6808, pp. 147-157.]

Keywords: Words separation, Finite automata

1. Introduction

In 1986, Goralčík and Koubek [1] introduced the *separating words problem*. Given two distinct strings w and x, we define $\operatorname{sep}(w, x)$ to be the number of states in the smallest deterministic finite automaton (DFA) that accepts w and rejects x [2]. This problem asks for good upper and lower bounds on

$$S(n) := \max_{w \neq x \wedge |w|, |x| \leq n} \operatorname{sep}(w, x).$$

Goralčík and Koubek [1] proved $S(n) = o(n)$. Besides, the best known upper bound so far is $O(n^{2/5} (\log n)^{3/5})$, which was obtained by Robson [3, 4]. A recent paper by Demaine, Eisenstat, Shallit, and Wilson [2] surveys the latest results about this problem, and while proving several new theorems, it also introduces three new open problems, all of which have remained unsolved

Email address: `febrahimnejad@ce.sharif.edu` (Farzam Ebrahimnejad)

until now. In this paper, we solve the first open problem stated in that paper, which asks whether

$$\left|\mathrm{sep}(w, x) - \mathrm{sep}(w^R, x^R)\right|$$

is bounded or not. We prove that this difference is actually unbounded. In order to do so, in Theorem 26 in subsection 2.5, for all positive integers k and n, we will construct two strings

$$w = u0^n v, x = u0^{n+(2n+1)!}v,$$

for some $u, v \in \{01, 11\}^+ \left(0^+ \{01, 11\}^+\right)^*$, such that $\mathrm{sep}(w, x) - \mathrm{sep}(w^R, x^R)$ approaches infinity as k and n approach infinity. As we will later see in Lemma 25 in subsection 2.4, under certain conditions, we can set u, v so that it requires relatively few states to separate w^R, x^R. But while preserving these conditions, by using the function C_n and the regular language G_k, which we will introduce in subsections 2.2 and 2.3, respectively, we can set u, v so that it will require exponentially more states, with respect to k, to separate w and x. We will see how exactly to do so in the rest of the paper.

2. Results

2.1. Preliminaries

We assume the reader is familiar with the basic concepts and terminology of automata theory as in, for example, [5]. In this subsection, we present some definitions and notation, and prove a few simple lemmas which will be used in the subsequent subsections.

In this paper, we let $\mathbb{N}$ denote the set of natural numbers, excluding 0.

Definition 1. We denote a DFA D by a 5-tuple $(Q_D, \Sigma, \delta_D, q_0, F_D)$, where Q_D is the set of states of D, Σ is the alphabet that D is defined over, δ_D is the transition function, $q_0 \in Q_D$ is the start state, and $F_D \subseteq Q_D$ is the set of accept states of D.

- For a state $q \in Q_D$ and a string $w \in \Sigma^*$, we define $\delta_D(q, w)$ to be the state in Q_D at which we end if we start reading w from q. Also, we define $\delta_D(w) \coloneqq \delta_D(q_0, w)$. We say that D accepts w if $\delta_D(w) \in F_D$, and otherwise we say that it rejects w. Moreover, for a subset of states $S \subseteq Q_D$ and a language $L \subseteq \Sigma^*$, we define

  $$\delta_D(S, L) \coloneqq \{q' \in Q_D \mid \exists q \in S, x \in L : q' = \delta_D(q, x)\}.$$

  Finally, we define $\delta_D(q, L) \coloneqq \delta_D(\{q\}, L)$.

- For a positive integer i, we define M_i to be the set of all DFAs E defined over $\{0, 1, 2\}$, where $|Q_E| \leq i$. Clearly, M_i is finite.
- In this paper, we assume $\Sigma = \{0, 1, 2\}$, unless stated otherwise explicitly.

Definition 2. Given a DFA D and two distinct strings $w, x \in \Sigma^*$, we say D *separates* two strings w and x, if it accepts w but rejects x. Now we can define $\operatorname{sep}(w, x)$ as the minimum number of states required for a DFA to separate w and x. Also, we say that D *distinguishes* w and x if $\delta_D(w) \neq \delta_D(x)$.

Notice that if a DFA separates two strings, then it must also distinguish them. The following simple lemma shows that a stronger connection exists between these two definitions.

Lemma 3. *For any two arbitrary strings $w, x \in \Sigma^*$, if a DFA D distinguishes w and x, then* $\operatorname{sep}(w, x) \leq |Q_D|$.

Proof. If some DFA D distinguishes two strings $w, x \in \Sigma^*$, then the DFA with the same set of states and transition function as D, but with $\delta_D(w)$ as the only accepting state separates w and x. Therefore we get $\operatorname{sep}(w, x) \leq |Q_D|$. $\square$

The following lemma shows that adding the same prefix and suffix to two distinct strings will not make it easier to separate them.

Lemma 4. *For any four strings $w, x, u, v \in \Sigma^*$ such that $w \neq x$, we have* $\operatorname{sep}(uwv, uxv) \geq \operatorname{sep}(w, x)$.

Proof. Let D be a DFA with $\operatorname{sep}(wv, xv)$ states that separates wv and xv. This DFA must distinguish w and x, so by Lemma 3 we have

$$\operatorname{sep}(w, x) \leq |Q_D| = \operatorname{sep}(wv, xv).$$

Besides, if some DFA E separates uwv and uxv, then the DFA with the same set of states and transitions as E but with $\delta_E(u)$ as the start state separates wv and xv. Hence we have

$$\operatorname{sep}(uwv, uxv) \geq \operatorname{sep}(wv, xv) \geq \operatorname{sep}(w, x).$$

$\square$

The next observation will be used several times throughout this paper, both in Lemma 9 and Theorem 26.

Proposition 5. *Let R be a regular language. If $x, y \in R\,(0^+R)^*$, then $x0^jy \in R\,(0^+R)^*$ for every positive integer j.*

Now let us consider the transitions on symbol 0. The following definition and proposition help us in the proof of Lemma 9 in the next subsection.

Definition 6. Assume D is a DFA over $\{0, 1, 2\}$. For a state $q \in Q_D$, we say q is in a *zero-cycle*, if there exists some positive integer i such that $\delta_D(q, 0^i) = q$. We call the minimum such i the length of this cycle.

Also, for a non-negative integer i, we define

$$\text{0-Path}_D(q, i) \coloneqq \left\{p = \delta_D(q, 0^j) \mid 0 \le j \le i \text{ and } p \text{ is not in a zero-cycle}\right\}.$$

Finally, we denote $\text{0-Path}_D(q, |Q_D|)$ by $\text{0-Path}_D(q)$.

Notice that if a state $\delta_D(q, 0^i)$ is in a zero-cycle, then for every j with $j > i$, the state $\delta_D(q, 0^j)$ is also in a zero-cycle. Using this fact, we get the following observation.

Proposition 7. *Let $D = (Q, \Sigma, \delta, q_0, F)$ be a DFA and i be a positive integer. For convenience, we will drop the subscript D from* 0-Path$_D$. *Then*

(a) $|\text{0-Path}(q, i)| \le i + 1$ *and* $|\text{0-Path}(q, i)| \le |\text{0-Path}(q)|$.

(b) If $\delta(q, 0^i)$ is not in a zero-cycle, then $|\text{0-Path}(q, i)| = i + 1 \le |Q|$.

(c) $|\text{0-Path}(q)| = |\text{0-Path}(q, i - 1)| + |\,|\text{0-Path}(\delta(q, 0^i))|$.

Proof. (a) and (b) follow directly from the definition and the fact above. To prove (c), notice that $\text{0-Path}(q, i - 1) \cap \text{0-Path}(\delta(q, 0^i)) = \emptyset$. □

2.2. The Strings f_n and g_n, and the Function C_n

As explained in the Introduction section, our goal is to find some strings u and v, so that by setting $w = u0^nv$ and $x = u0^{n+(2n+1)!}v$, $\text{sep}(w, x)$ becomes arbitrarily greater than $\text{sep}(w^R, x^R)$. The purpose of this subsection is to help us set u and v so that $\text{sep}(w, x)$ becomes large enough. Actually, it is not hard to show that $\text{sep}(0^n, 0^{n+(2n+1)!}) = n + 2$. By Lemma 4, it follows that regardless of what u and v are, the values $\text{sep}(w, x)$ and $\text{sep}(w^R, x^R)$ are at least $n + 2$. In Lemma 9, we show that we can set u and v so that $\text{sep}(w, x) \ge 2n + 2$. However, this lemma does not guarantee a low value for $\text{sep}(w^R, x^R)$, and so Lemma 9 alone does not solve the problem. But still, it plays a crucial role in the proof of Theorem 26 in subsection 2.5, and in the next subsections, we will see how to fix this issue.

Definition 8. Since 0^n and $0^{n+(2n+1)!}$ are used frequently throughout this paper, from now on, for convenience, we denote them by f_n and g_n, respectively.

Lemma 9. *For all $n \in \mathbb{N}$ and $w_0 \in \Sigma^+$, there exists $w \in w_0(0^+w_0)^*$ such that* $\operatorname{sep}(wf_nw, wg_nw) \geq 2n+2$. *We denote the w corresponding to w_0 by $C_n(w_0)$.*

Proof. We run the following algorithm iteratively, while increasing i by 1 at each step, starting from $i = 1$. While running this algorithm, we preserve the condition that $w_i \in w_0(0^+w_0)^*$. Obviously this condition holds when $i = 0$.

In each iteration, if there exists a DFA $D = (Q, \Sigma, \delta, q_0, F) \in M_{2n+1}$ such that $\delta(v, 0^y w_{i-1}) = \delta(v', 0^y w_{i-1})$ for some distinct states $v, v' \in \delta(Q, w_{i-1})$ and some positive integer y, then we set

$$w_i = w_{i-1}0^y w_{i-1}.$$

Otherwise, we set $w_i = w_{i-1}$ and terminate. By the loop condition stated above, we have $w_{i-1} \in w_0(0^+w_0)^*$. Therefore by Proposition 5, $w_i \in w_0(0^+w_0)^*$ and hence the loop condition holds for w_i. Furthermore, let E be an arbitrary DFA in M_{2n+1}. Since w_{i-1} is a prefix of w_i, if for two states $s, s' \in Q_E$ we have $\delta_E(s, w_{i-1}) = \delta_E(s', w_{i-1})$, then $\delta_E(s, w_i) = \delta_E(s', w_i)$. Therefore we have $|\delta_E(Q_E, w_i)| \leq |\delta_E(Q_E, w_{i-1})|$. Moreover, by the choice of v and v' it follows that $|\delta(Q, w_i)| < |\delta(Q, w_{i-1})|$. Hence we can write

$$\sum_{E \in M_{2n+1}} |\delta_E(Q_E, w_i)| < \sum_{E \in M_{2n+1}} |\delta_E(Q_E, w_{i-1})| .$$

Thus $\sum_{E \in M_{2n+1}} |\delta_E(Q_E, w_i)|$ decreases by at least one in each step, and therefore, this algorithm terminates after a finite number of iterations. Suppose it terminates after l iterations. We set $w = w_l$.

Now we claim $\operatorname{sep}(wf_nw, wg_nw) \geq 2n+2$. We prove by backward induction on t that for all $t \geq n$, no DFA in M_{2n+1} can distinguish $w0^tw$ and $w0^t0^{(2n+1)!}w$. In other words, we will prove by induction on t that for all integers $t \geq n$, DFAs $D \in M_{2n+1}$, and states $q \in \delta(Q, w)$, we have

$$\delta(q, 0^t w) = \delta(q, 0^t 0^{(2n+1)!} w).$$

Base step: Consider $t \geq 2n+1$. Let $D = (Q, \Sigma, \delta, q_0, F)$ be an arbitrary DFA in M_{2n+1}. For all states $q \in \delta(Q, w)$, the state $\delta(q, 0^t)$ must be in a zero-cycle because otherwise by Proposition 7, we have

$$|Q| \geq |0\text{-Path}(q, t)| = t + 1 \geq 2n+2,$$

which is a contradiction. Since the size of the zero-cycle containing $\delta(q, 0^t)$ is at most $|Q| \leq 2n+1$, it divides $(2n+1)!$. Thus $\delta(q, 0^t 0^{(2n+1)!}) = \delta(q, 0^t)$, and hence we have $\delta(q, 0^t w) = \delta(q, 0^t 0^{(2n+1)!} w)$.

Induction step: Consider $n \leq t < 2n+1$. By the induction hypothesis we know that the claim holds for all $t' > t$. Let $D = (Q, \Sigma, \delta, q_0, F)$ be an arbitrary DFA in M_{2n+1}. For convenience, we will drop the subscript D from 0-Path$_D$. Pick one of the states $q \in \delta(Q, w)$ maximizing $|\text{0-Path}(q)|$ amongst all members of $\delta(Q, w)$. First, we prove the claim for all $p \in \delta(Q, w)$ where $p \neq q$. If $\delta(p, 0^t)$ is in a zero-cycle, then by a similar argument as in the base case, we obtain $\delta(p, 0^t w) = \delta(p, 0^t 0^{(2n+1)!} w)$, and the proof is complete. Now suppose that $\delta(p, 0^t)$ is not in a zero-cycle. By Proposition 7, we have $|\text{0-Path}(p)| \geq |\text{0-Path}(p, t)| = t+1$. Therefore by the choice of q, we have $|\text{0-Path}(q)| \geq |\text{0-Path}(p)| \geq t+1$. Hence $\delta(q, 0^t) \in \text{0-Path}(q)$, and so it is not in a zero-cycle. Thus, we have $|\text{0-Path}(q, t)| = t+1$. If $\text{0-Path}(q, t) \cap \text{0-Path}(p, t) = \emptyset$, then since $\text{0-Path}(q, t)$ and $\text{0-Path}(p, t)$ are subsets of Q, we see that

$$|Q| \geq |\text{0-Path}(q, t)| + |\text{0-Path}(p, t)| = (t+1) + (t+1) = 2t + 2 \geq 2n + 2,$$

which is a contradiction. So there exists some $r \in \text{0-Path}\,(q, t) \cap \text{0-Path}\,(p, t)$. By definition, there exist $0 \leq a, b \leq t$ such that $\delta(q, 0^a) = \delta(p, 0^b) = r$. The state r is not in a zero-cycle because otherwise, since $t \geq b$, $\delta(p, 0^t)$ should also be in a zero-cycle, which contradicts our assumption. Hence by Proposition 7, we have

$$\begin{aligned} |\text{0-Path}(q)| &= |\text{0-Path}(q, a-1)| + |\text{0-Path}\,(\delta(q, 0^a))| \\ &= a + |\text{0-Path}(r)|\,, \end{aligned} \tag{1}$$

and similarly, we get

$$|\text{0-Path}(p)| = b + |\text{0-Path}(r)|\,. \tag{2}$$

By subtracting equation 2 from equation 1, we obtain

$$|\text{0-Path}(q)| - |\text{0-Path}(p)| = a - b.$$

But we have $|\text{0-Path}(q)| \geq |\text{0-Path}(p)|$. Hence $a \geq b$.

Suppose $a = b$. Then $\delta(q, 0^a) = \delta(p, 0^a)$. So if $a = 0$, then it follows that $p = q$, which contradicts our assumption. Therefore $a > 0$. We have

$\delta(q, 0^a) = \delta(p, 0^a)$, so $\delta(q, 0^a w) = \delta(p, 0^a w)$. Hence the algorithm could not have terminated, which is a contradiction. Thus we have $a > b$, and so by the induction hypothesis for $(a-b)+t > t$, we have

$$\delta(q, 0^{(a-b)+t} w) = \delta(q, 0^{(a-b)+t} 0^{(2n+1)!} w). \tag{3}$$

But since $b \leq t$, we obtain

$$\delta(q, 0^{(a-b)+t}) = \delta(q, 0^a 0^{t-b}) = \delta(r, 0^{t-b}) = \delta(p, 0^b 0^{t-b}) = \delta(p, 0^t). \tag{4}$$

By equations 3 and 4 we get

$$\delta(p, 0^t w) = \delta(q, 0^{(a-b)+t} w) = \delta(q, 0^{(a-b)+t} 0^{(2n+1)!} w) = \delta(p, 0^t 0^{(2n+1)!} w),$$

and therefore the proof is complete for p.

It only remains to prove the claim for q. Let us write

$$A = \delta\left(\delta(Q, w) - \{q\}, 0^t w\right)$$

and

$$B = \delta\left(\delta(Q, w) - \{q\}, 0^t 0^{(2n+1)!} w\right).$$

We know for any two distinct states $s, s' \in \delta(Q, w)$, we have

$$\delta(s, 0^t w) \neq \delta(s', 0^t w)$$

and

$$\delta(s, 0^t 0^{(2n+1)!} w) \neq \delta(s', 0^t 0^{(2n+1)!} w)$$

because otherwise the algorithm could not have terminated, which is a contradiction. So $|A| = |B| = |\delta(Q, w)| - 1$. But we proved the induction step for all members of $\delta(Q, w)$ except q. Hence for all states $s \in \delta(Q, w) - \{q\}$ we have $\delta(s, 0^t w) = \delta(s, 0^t 0^{(2n+1)!} w)$. Therefore $A = B$. Let us write $e = \delta(q, 0^t w)$ and $e' = \delta(q, 0^t 0^{(2n+1)!} w)$. Since w is a suffix of both $0^t w$ and $0^t 0^{(2n+1)!} w$, by definition we have $e, e' \in \delta(Q, w)$. Also, since the algorithm has terminated, we get $e \notin A$ and $e' \notin B$. Consequently we have

$$e \in \delta(Q, w) - A,$$

and

$$e' \in \delta(Q, w) - B = \delta(Q, w) - A.$$

But since w is a suffix of $0^t w$, we have $A \subseteq \delta(Q, w)$. So

$$|\delta(Q, w) - A| = |\delta(Q, w)| - (|\delta(Q, w)| - 1) = 1.$$

Therefore $e = e'$ and the proof is complete. $\square$

2.3. The Regular Language G_k

In this subsection, we introduce the regular language $G_k \subseteq \{1,2\}^*$, which has some interesting characteristics. For all $k \in \mathbb{N}$, there exists a DFA with $O(k)$ states that accepts G_k^R, while no DFA with less than 2^k states accepts G_k. Similar regular languages that also have these two characteristics have been defined before [6, 7, 8] but are not quite appropriate for our purposes. Another characteristic of G_k is that, as proven later in Lemma 17, there exists $z_k \in G_k$ such that if a DFA with less than 2^k states accepts z_k, then it should also accept some string in $\{1,2\}^* - G_k$. This, together with Lemma 9, helps us construct the desired strings in Theorem 26. Recall that $\mathbb{N}$ denotes the set of positive integers.

Definition 10. For every positive integer k, we define languages L_k and G_k over $\{1,2\}$ as follows:

$$\begin{aligned} L_k \coloneqq & \left\{1^{2i}2 \mid i \in \mathbb{N} \wedge i \leq k\right\} \\ & \cup \left\{1^{i_1}21^{i_2}2\cdots 21^{i_{s-1}}21^{i_s}2 \quad \mid s, i_1, i_2, \ldots, i_s \in \mathbb{N} \right. \\ & \qquad \qquad \qquad \qquad \qquad \quad \wedge i_1 + i_2 + \cdots + i_s = 2k+1 \\ & \qquad \qquad \qquad \qquad \qquad \quad \left. \wedge i_1, i_2, \ldots, i_{s-1} \equiv 0 \pmod 2\right\}. \end{aligned}$$

Finally, we define $G_k \coloneqq L_k^*$

Lemma 11. *For all $u, v \in \Sigma^*$, we have $u1^{2k+1}2v \in G_k$ if and only if $u, v \in G_k$.*

Proof. We can easily observe that if $x1^{2k+1}2y \in L_k$, then $x = y = \epsilon$. Thus it follows that if $u1^{2k+1}2v \in L_k^* = G_k$, then both u and v should also be in G_k.

For the other direction, obviously we have $1^{2k+1}2 \in L_k$. Therefore by definition, if $u, v \in G_k$ then $u1^{2k+1}2v \in G_k$. $\square$

Definition 12. For a regular language $L \subseteq \Sigma^*$, we define $\text{sc}(L)$, or the state complexity of L, to be the minimum number of states required for a DFA to accept L. This concept has been studied for a long time; see, for example, [9, 10, 11].

Lemma 13. *For all integers $k \in \mathbb{N}$, we have $\text{sc}(G_k) \geq 2^k$.*

Proof. Let E be the set of all positive even numbers less than $2k+1$ and $\mathbb{P}(E)$ be the set of all subsets of E. We define the function $r : \mathbb{P}(E) \to \{1,2\}^*$ as follows:

For the empty set, we define $r(\emptyset) \coloneqq \epsilon$. Now consider an arbitrary non-empty subset of E, such as

$$S = \{a_1, a_2, \ldots, a_m\}.$$

Without loss of generality, assume $a_1 < a_2 < \cdots < a_m$. We define

$$r(S) \coloneqq 1^{a_m - a_{m-1}} 2 1^{a_{m-1} - a_{m-2}} 2 \cdots 2 1^{a_2 - a_1} 2 1^{a_1} 2.$$

Let $1 \le i < 2k+1$ be an odd number and $S \subseteq E$. We claim $r(S)1^i 2 \in G_k$ if and only if $2k+1-i \in S$. If $w = r(S)1^i 2 \in G_k$, then by definition $x \in G_k$ and $y \in L_k$ exist such that $w = xy$. But i is an odd number and $1^i 2$ is a suffix of y. Therefore by definition, $1 \le p \le m$ and $b_1, b_2, \ldots, b_p \in \mathbb{N}$ exist such that

$$b_p + b_{p-1} + \cdots + b_1 + i = 2k+1,$$

and

$$y = 1^{b_p} 2 1^{b_{p-1}} 2 \cdots 1^{b_1} 2 1^i 2.$$

It follows that

$$b_1 = a_1, b_2 = a_2 - a_1, \ldots, b_p = a_p - a_{p-1}.$$

Thus

$$a_p = b_1 + \cdots + b_p = 2k+1-i.$$

So $2k+1-i \in S$.

For the other direction, suppose $2k+1-i \in S$. Then $a_j \in S$ exists such that $a_j = 2k+1-i$. All members of S are even numbers less than $2k+1$. Hence, by definition we have

$$1^{a_m - a_{m-1}} 2, 1^{a_{m-1} - a_{m-2}} 2, \ldots, 1^{a_{j+1} - a_j} 2 \in L_k.$$

Moreover, we have

$$i + (a_1 + \sum_{t=2}^{j} a_t - a_{t-1}) = (2k+1-a_j) + a_j = 2k+1.$$

Hence

$$1^{a_j - a_{j-1}} 2 \cdots 1^{a_2 - a_1} 2 1^{a_1} 2 1^i 2 \in L_k.$$

Therefore $r(S)1^i 2 \in G_k = L_k^*$.

Now consider the family of strings $\{r(S) \mid S \in \mathbb{P}(E)\}$ of size 2^k. Let S and S' be two distinct sets in this family. Then, without loss of generality, there is an integer c with $c \in S \setminus S'$. Therefore, by our claim above, we have $r(S)1^{2k+1-c} \in G_k$ while $r(S')1^{2k+1-c} \notin G_k$. It follows that $\mathrm{sc}(G_k) \geq 2^k$. $\square$

Lemma 14. *For all integers $k \in \mathbb{N}$, we have* $\mathrm{sc}(G_k^R) \leq 5k+3$.

Proof. It suffices to prove there exists a DFA $D \in M_{5k+3}$ such that $L(D) = G_k^R$. We define $D = (Q, \Sigma, \delta, p_{2k+1}, F)$ as follows:

We set

$$Q = \{p_i \mid 0 \leq i \leq 2k+1\} \cup \{r_i \mid 2 \leq i \leq 2k+1\} \cup \left\{r'_{2i-1} \mid 1 \leq i \leq k\right\} \cup \{d\}$$

Additionally, we specify the following rules for the transition function:

- $\delta(p_i, 1) = p_{i+1}$ $(0 \leq i \leq 2k)$,
- $\delta(r_i, 1) = r_{i+1}$ $(2 \leq i \leq 2k)$,
- $\delta(r_{2i-1}, 2) = r'_{2i-1}$ $(2 \leq i \leq k)$,
- $\delta(r'_{2i-1}, 1) = r_{2i}$ $(1 \leq i \leq k)$,
- $\delta(p_i, 2) = \begin{cases} p_0, & \text{if } 2 \leq i \leq 2k \text{ and } i \text{ is even;} \\ r'_i, & \text{if } 1 \leq i \leq 2k \text{ and } i \text{ is odd,} \end{cases}$
- $\delta(p_{2k+1}, 2) = \delta(r_{2k+1}, 2) = p_0$,

and all the remaining transitions go to the dead state d. The DFA D is shown in Figure 1.

Finally, we set

$$F = \{p_{2i} \mid 1 \leq i \leq k\} \cup \{p_{2k+1}, r_{2k+1}\}.$$

It is not hard to verify that $\delta(F, L_k^R) \subseteq F$, and hence $\delta(F, \left(L_k^R\right)^*) = \delta(F, G_k^R) \subseteq F$. It is also easy to show that $\delta(F, \Sigma^* - G_k^R) \cap F = \emptyset$. Thus since $p_{2k+1} \in F$, we obtain $L(D) = G_k^R$. Therefore we have $\mathrm{sc}(G_k^R) \leq |Q| = 5k+3$. $\square$

Definition 15. For $w \in \Sigma^*$ and a language L over Σ, we define $\mathrm{lsep}(w, L)$ as the minimum number of states of a DFA that accepts w and rejects all $x \in L$.

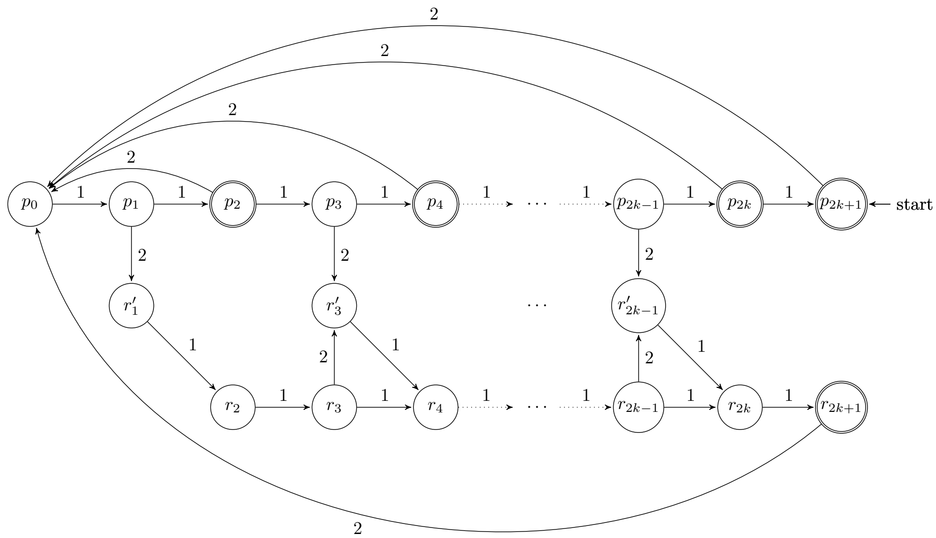

Figure 1: The DFA D which is explained in Lemma 14. The reject state d is not shown.

Definition 16. Since the set $\{1,2\}^* - G_k$ is referred to several times in the rest of this paper, for simplicity, we will denote it by H_k.

Lemma 17. *There exists* $z_k \in (G_k - \{\epsilon\})$ *such that* $\text{lsep}(z_k, H_k) \geq 2^k$.

Proof. At the beginning, we set $w_0 = \epsilon$, $U_0 = M_{2^k-1}$, and $V_0 = \emptyset$. We preserve the following conditions for all $j \geq 0$:

1. $V_j \cup U_j = M_{2^k-1}$;
2. $w_j \in G_k$;
3. For all DFAs $D \in V_j$, there exists some $r \in H_k$ such that D does not distinguish r and w_j.

Obviously these conditions hold for $j = 0$.

Now we run the following algorithm iteratively, while increasing i by 1 at each step, starting with $i = 1$:

In each iteration, if there exists a DFA $D = (Q, \Sigma, \delta, q_0, F) \in U_{i-1}$, and strings $x \in G_k$ and $y \in H_k$ such that $\delta(w_{i-1}1^{2k+1}2x) = \delta(w_{i-1}1^{2k+1}2y)$, then we set $w_i = w_{i-1}1^{2k+1}2x$, $U_i = U_{i-1} - \{D\}$, and $V_i = V_{i-1} \cup \{D\}$. Otherwise, we terminate by setting $w_i = w_{i-1}$, $U_i = U_{i-1}$ and $V_i = V_{i-1}$.

Obviously, Condition 1 holds for $j = i$. Moreover, by Condition 2 for $j = i - 1$, we have $w_{i-1} \in G_k$. Therefore by Lemma 11, we have $w_i \in G_k$, and hence Condition 2 holds for $j = i$.

Furthermore, by Condition 3 for $j = i - 1$, for all DFAs $E \in V_{i-1}$, there exists $r \in H_k$ such that $\delta_E(r) = \delta_E(w_{i-1})$. Hence we have

$$\delta_E(r1^{2k+1}2x) = \delta_E(w_{i-1}1^{2k+1}2x) = \delta_E(w_i).$$

But by Lemma 11, we obtain $r1^{2k+1}2x \in H_k$. Thus Condition 3 for $j = i$ holds for all members of $V_{i-1} = V_i - \{D\}$. It only remains to prove that it also holds for D. We have

$$\delta(w_i) = \delta(w_{i-1}1^{2k+1}2x) = \delta(w_{i-1}1^{2k+1}2y).$$

But by Lemma 11, we get $w_{i-1}1^{2k+1}2y \in H_k$. Hence Condition 3 holds for D. Therefore Condition 3 holds for $j = i$.

This algorithm terminates after a finite number of iterations because $|U_i|$ decreases by 1 at each step (except the last one). Suppose it terminates after t iterations. We claim $U_t = \emptyset$. Otherwise there exists some DFA $B \in U_t$. Let B' be the DFA with the same set of states and transition function as B, but with $\delta_B(w_{t-1}1^{2k+1}2)$ as the start state and with $\delta_B(\delta_B(w_{t-1}1^{2k+1}2), G_k)$ as the set of accepting states. By definition, we have $G_k \subseteq L(B')$. Furthermore, B' cannot accept any string $w \notin G_k$ because otherwise

$$\delta_B(\delta_B(w_{t-1}1^{2k+1}2), G_k) \cap \delta_B(\delta_B(w_{t-1}1^{2k+1}2), H_k) \neq \emptyset,$$

and therefore the algorithm could not have terminated, which is a contradiction. Hence $L(B') = G_k$. So by Lemma 13 we have $B' \notin M_{2^k-1}$, which contradicts $|Q_{B'}| = |Q_B| \leq 2^k - 1$.

Thus by Condition 1 it follows that $V_t = M_{2^k-1}$. By Condition 3, for all DFAs $D \in M_{2^k-1}$, if D accepts w_t, then it also accepts some string in H_k. Hence we obtain $\operatorname{lsep}(w_t, H_k) \geq 2^k$. By Condition 2, we have $w_t \in G_k$. Since V_t is not empty, we obtain that the algorithm has terminated after a positive number of iterations. Furthermore, for $1 \leq i \leq t$, the string w_i starts with $1^{2k+1}2$. Hence w_t is not empty, and so we have $w_t \in (G_k - \{\epsilon\})$. Therefore we can set $z_k \coloneqq w_t$. $\square$

Definition 18. The set $H_k \cup \{z_k\}$ is referred to several times in the rest of this paper. So, for simplicity, we will denote it by H'_k.

Remark 19. For any two DFAs $D \in M_i$ and $D' \in M_j$, some DFA $E \in M_{i \times j}$ exists such that $L(E) = L(D) \cap L(D')$.

Lemma 20. *For every two DFAs $D, D' \in M_{2^{k/2}-1}$, and every string $w \in \Sigma^*$, there exists $x \in H_k$ such that $\delta_D(wz_k) = \delta_D(wx)$ and $\delta_{D'}(wz_k) = \delta_{D'}(wx)$.*

Proof. Let $D = (Q, \Sigma, \delta, q_0, F)$ and $D' = (Q', \Sigma, \delta', q_0', F')$. We set E to be the same DFA as D but with $\delta(w)$ as the start state, and with $\delta(wz_k)$ as the only accept state. Similarly, we set E' to be the same DFA as D' but with $\delta'(w)$ as the start state, and with $\delta'(wz_k)$ as the only accept state. Obviously, $z_k \in L(E) \cap L(E')$. By Lemma 17, $L(E) \cap H_k$ and $L(E') \cap H_k$ are not empty. We further claim that their intersection, $L(E) \cap L(E') \cap H_k$, is also not empty. Otherwise, by Remark 19, some DFA F with at most

$$(2^{k/2} - 1)(2^{k/2} - 1) < 2^k$$

states exists such that $L(F) = L(E) \cap L(E')$. If $L(E) \cap L(E') \cap H_k = \emptyset$, then we obtain $L(F) \cap H_k = \emptyset$. But $z_k \in L(F)$. Hence F accepts z_k but rejects every string in H_k, and therefore, by Lemma 17, we have $|Q_F| \geq 2^k$, which is a contradiction. Hence there exists some

$$x \in L(E) \cap L(E') \cap H_k,$$

or equivalently, both E and E' accept some $x \in H_k$. Furthermore, by the construction of E and E', we obtain $\delta(wz_k) = \delta(wx)$ and $\delta'(wz_k) = \delta'(wx)$, and therefore the proof is complete. □

Lemma 21. *Let $w \in H_k' \, (0^+ H_k')^*$. For any two DFAs $D, D' \in M_{2^{k/2}-1}$, there exists some $w' \in H_k \, (0^+ H_k)^*$ such that $\delta_D(w) = \delta_D(w')$ and $\delta_{D'}(w) = \delta_{D'}(w')$, or in other words, neither D nor D' distinguishes w and w'.*

Proof. We have $w \in H_k' \, (0^+ H_k')^*$. So it can be expressed as

$$w = u_1 0^{i_1} \cdots u_{l-1} 0^{i_{l-1}} u_l 0^{i_l} u_{l+1},$$

where $i_1, \ldots, i_l \in \mathbb{N}$ and $u_1, \ldots, u_{l+1} \in H_k'$. For $1 \leq j \leq l$, let us write

$$w_j = u_1 0^{i_1} \cdots u_{j-1} 0^{i_{j-1}} u_j 0^{i_j}.$$

For simplicity, we also set $i_0 = 0$ and $w_0 = \epsilon$. Now for $1 \leq j \leq l+1$, we define the strings $u_j' \in H_k$ as follows: If $u_j \neq z_k$, then we set $u_j' = u_i$.

Otherwise, let $D = (Q, \Sigma, \delta, q_0, F)$ and $D' = (Q', \Sigma, \delta', q'_0, F')$. By Lemma 20, it follows that there exists $x \in H_k$ such that $\delta(w_{j-1}z_k) = \delta(w_{j-1}x)$ and $\delta'(w_{j-1}z_k) = \delta'(w_{j-1}x)$. We set $u'_j = x$. In either of the cases, clearly we have $u'_j \in H_k$. Now let us write

$$w' = u'_1 0^{i_1} \cdots u'_l 0^{i_l} u'_{l+1}.$$

We claim $\delta(w) = \delta(w')$ and $\delta'(w) = \delta'(w')$. Let us set $x_0 = x'_0 = \epsilon$. Moreover, for $1 \le j \le l+1$, we set

$$x_j = u_1 0^{i_1} \cdots u_{j-1} 0^{i_{j-1}} u_j,$$

and

$$x'_j = u'_1 0^{i_1} \cdots u'_{j-1} 0^{i_{j-1}} u'_j.$$

We prove by induction that for $0 \le j \le l+1$, we have $\delta(x_j) = \delta(x'_j)$ and $\delta'(x_j) = \delta'(x'_j)$. The base step is obvious for $j = 0$. For $j \ge 1$, if $u_j \ne z_k$, then we have $u'_j = u_j$, and so we can obtain the claim. Otherwise, by the induction hypothesis we have $\delta(x_{j-1}) = \delta(x'_{j-1})$. By the choice of u'_{j-1} we have

$$\begin{aligned} \delta(x_j) = \delta(w_{j-1}z_k) = \delta(w_{j-1}u'_j) \quad &= \delta(x_{j-1}0^{i_{j-1}}u'_j) \\ &= \delta(x'_{j-1}0^{i_j-1}u'_j) = \delta(x'_j), \end{aligned}$$

and the proof of the claim is complete. Similarly, we can prove $\delta'(x_j) = \delta'(x'_j)$ for all $0 \le j \le l+1$. Hence we obtain

$$\delta(w) = \delta(x_{l+1}) = \delta(x'_{l+1}) = \delta(w')$$

and

$$\delta'(w) = \delta'(x_{l+1}) = \delta'(x'_{l+1}) = \delta'(w').$$

Besides, for all $1 \le j \le l+1$, we have $u'_j \in H_k$. Therefore it follows that $w' \in H_k\,(0^+H_k)^*$, and hence the proof is complete.

□

Proposition 22. *Let D be a DFA in $M_{2^{k/2}-1}$, $q, q' \in Q_D$, and $w \in H'_k\,(0^+H'_k)^*$. There exists some $w' \in H_k\,(0^+H_k)^*$ such that $\delta_D(q, w) = \delta_D(q, w')$ and $\delta_D(q', w) = \delta_D(q', w')$.*

Proof. We define two new DFAs E and E', having the same set of states and transition function as D, but with q and q' as their starting states, respectively. The proposition follows directly from applying Lemma 21 to E, E' and w. □

2.4. Mapping $\{0,1,2\}^*$ *to* $\{0,1\}^*$

The previous lemmas may help us to construct two strings in $\Sigma^* = \{0,1,2\}^*$ with our desired characteristics. But our goal is to prove our result for an alphabet of size 2. To be able to construct the intended strings over $\{0,1\}$, in this subsection we introduce the function tr that maps strings in Σ^* to strings in $\{0,1\}^*$, while preserving some of our desired characteristics in them.

Definition 23. For a string $w \in \Sigma^*$, we define $\mathrm{tr}(w)$ to be the string obtained from w by replacing all occurrences of 1 by 11 and all occurrences of 2 by 10. Clearly we have $\mathrm{tr}(w) \in \{0,1\}^*$

The following lemma shows that when two strings are mapped under tr^R, separating them would be at least as hard as separating the original ones.

Lemma 24. *For all pairs of distinct strings* $w, x \in \Sigma^*$*, we have*

$$\mathrm{sep}(\mathrm{tr}^R(w), \mathrm{tr}^R(x)) \geq \mathrm{sep}(w, x).$$

Proof. Let $D = (Q, \Sigma, \delta_D, q_0, F)$ be a DFA that separates $\mathrm{tr}^R(w)$ and $\mathrm{tr}^R(x)$. We construct a new DFA $E = (Q, \Sigma, \delta_E, q_0, F)$ that separates w and x. For all states $q \in Q$, we set

$$\delta_E(q,0) = \delta_D(q,0), \delta_E(q,1) = \delta_D(q,11), \delta_E(q,2) = \delta_D(q,01).$$

It is fairly easy to see that for all strings $u \in \Sigma^*$, we have $\delta_E(u) = \delta_D(\mathrm{tr}^R(u))$. Since D separates $\mathrm{tr}^R(w)$ and $\mathrm{tr}^R(x)$, the DFA E separates w and x. □

Lemma 25. *Let* $t \in \mathbb{N}$ *and* $R \subseteq \{1,2\}^*$ *be a regular language such that* $\mathrm{sc}(R) \leq t$*. Also, let* $w \in ((\{1,2\}^* - R)\, 0^+)^* \,(R - \{\epsilon\})$*. For all* $w' \in 1\,\{0,1\}^*$*, we have*

$$\mathrm{sep}(\mathrm{tr}(w) f_n w', \mathrm{tr}(w) g_n w') \leq 2t + n + 4.$$

Recall that $f_n = 0^n$ *and* $g_n = 0^{n+(2n+1)!}$*.*

Proof. We have $\mathrm{sc}(R) \leq t$. So there exists a DFA $D = (Q_D, \Sigma, \delta_D, q_0, F_D) \in M_t$ such that $L(D) = R$. By using D, we construct another DFA $E \in M_{2t+n+4}$ that distinguishes $\mathrm{tr}(w) f_n w'$ and $\mathrm{tr}(w) g_n w'$. Assume

$$Q_D = \{q_0, q_1, \ldots, q_{m-1}\}$$

for some $m \leq t$. We then set $E = (Q, \Sigma, \delta, s, \emptyset)$, where

$$Q = Q_D \cup \left\{q'_0, q'_1, \ldots, q'_{m-1}, r_1, \ldots, r_n, r_{n+1}, s, p, p'\right\}.$$

Also, we specify the following rules for the transition function of E:

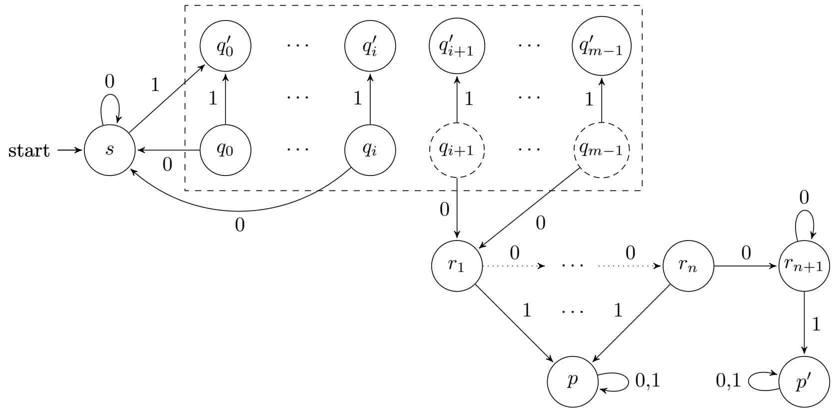

Figure 2: The DFA E, which is explained in Lemma 25 (assuming $q_{i+1}, \ldots, q_{m-1}$ are the only accept states in D).

- For $0 \leq i \leq m-1$, we set

$$\delta(q_i, 1) = q_i', \delta(q_i', 0) = \delta_D(q_i, 2), \delta(q_i', 1) = \delta_D(q_i, 1).$$

- For all $q_i \notin F_D$, we set $\delta(q_i, 0) = s$.
- For all $q_i \in F_D$, we set $\delta(q_i, 0) = r_1$.
- For $1 \leq i \leq n$, we set $\delta(r_i, 0) = r_{i+1}$ and $\delta(r_i, 1) = p$.
- $\delta(p, 0) = \delta(p, 1) = p$.
- $\delta(r_{n+1}, 0) = r_{n+1}$ and $\delta(r_{n+1}, 1) = \delta(p', 0) = \delta(p', 1) = p'$.
- $\delta(s, 0) = s$ and $\delta(s, 1) = q_0'$; see Figure 2 for an illustration.

Clearly, for all $0 \leq i \leq m-1$, we have $\delta(q_i, 11) = \delta_D(q_i, 1)$ and $\delta(q_i, 10) = \delta_D(q_i, 2)$. Hence for all $u \in \{1,2\}^*$, we have $\delta(\mathrm{tr}(u)) = \delta_D(u)$. Since $w \in ((\{1,2\}^* - R)\, 0^+)^* \,(R - \{\epsilon\})$, we have $\delta(\mathrm{tr}(w)) \in F_D$. Therefore we can show that $\delta(\mathrm{tr}(w) f_n w') = p$ and $\delta(\mathrm{tr}(w) g_n w') = p'$. Thus E distinguishes $\mathrm{tr}(w) f_n w'$ and $\mathrm{tr}(w) g_n w'$. So by Lemma 3 we get

$$\mathrm{sep}\,(\mathrm{tr}(w) f_n w', \mathrm{tr}(w) g_n w') \leq |Q| \leq 2t + n + 4.$$

$\square$

2.5. The Main Result

Now we are ready to prove our main result. As shown in Theorem 27, by substituting the appropriate values for n and k in Theorem 26, we can prove that the difference $\left|\text{sep}(w, x) - \text{sep}(w^R, x^R)\right|$ is unbounded.

Theorem 26. *For all $k, n \in \mathbb{N}$, there exist two unequal strings $w', x' \in \{0,1\}^*$ such that*

$$\text{sep}(w', x') \geq \min(2n + 2, 2^{k/2}),$$

but

$$\text{sep}((w')^R, (x')^R) \leq n + 10k + 10.$$

Proof. Let us write $p = \min(2n+2, 2^{k/2}) - 1$. Consider an arbitrary ordering of all pairs of DFAs in M_p and each of their states:

$$(D_1, s_1), (D_2, s_2), \ldots, (D_m, s_m),$$

where $s_i \in Q_{D_i}$, and m is the total number of such pairs, which is clearly finite. Here, for convenience, we use subscript i instead of D_i. So let $D_i = (Q_i, \Sigma, \delta_i, q_i, F_i)$.

We start with

$$u_0 = v_0 = z_k, w_0 = u_0 f_n v_0 = z_k f_n z_k, x_0 = u_0 g_n v_0 = z_k g_n z_k.$$

During the execution of the algorithm that we explain below, we preserve the following conditions for all $0 \leq e \leq m$:

1. $u_e \in H'_k \, (0^+ H'_k)^*$.
2. $v_e \in z_k \, (0^+ H_k)^*$. We have $z_k \in H'_k$ and $H_k \subset H'_k$. Therefore $v_e \in H'_k \, (0^+ H'_k)^*$.
3. $w_e = u_e f_n v_e$ and $x_e = u_e g_n v_e$.
4. For all $1 \leq j \leq e$ and $\alpha, \alpha' \in \Sigma^*$, if $\delta_j(\alpha u_e) = s_j$, then $\delta_j(\alpha w_e \alpha') = \delta_j(\alpha x_e \alpha')$. By setting $\alpha = \alpha' = \epsilon$, it follows that if $\delta_j(u_e) = s_j$ then $\delta_j(w_e) = \delta_j(x_e)$.

We can easily observe that these conditions hold for $e = 0$. Now we run the following algorithm iteratively for $i = 1, 2, \ldots, m$:

By Condition 3 for $e = i-1$, we have $w_{i-1} = u_{i-1} f_n v_{i-1}$ and $x_{i-1} = u_{i-1} g_n v_{i-1}$. We set

$$u_i = C_n(v_{i-1} 0 u_{i-1}),$$

where C_n is given by Lemma 9. By Lemma 9, we have

$$u_i \in v_{i-1} 0 u_{i-1} \left(0^+ v_{i-1} 0 u_{i-1}\right)^* .$$

So it can be expressed as

$$u_i = v_{i-1} 0 u_{i-1} 0^{i_1} v_{i-1} 0 u_{i-1} 0^{i_2} \cdots v_{i-1} 0 u_{i-1} 0^{i_l} v_{i-1} 0 u_{i-1},$$

for some $l, i_1, \ldots, i_l \in \mathbb{N}$.

By Conditions 1 and 2 for $e = i-1$, we have $u_{i-1}, v_{i-1} \in H'_k \left(0^+ H'_k\right)^*$. Hence by Proposition 5, we have $v_{i-1} 0 u_{i-1} \in H'_k \left(0^+ H'_k\right)^*$. Therefore by using Proposition 5 again, we get $u_i \in H'_k \left(0^+ H'_k\right)^*$. So Condition 1 holds for $e = i$.

Moreover, let us write

$$y = u_{i-1} 0^{i_1} v_{i-1} 0 u_{i-1} 0^{i_2} \cdots v_{i-1} 0 u_{i-1} 0^{i_l} v_{i-1} 0 u_{i-1}.$$

Clearly, we have $u_i = v_{i-1} 0 y$. With the same argument as for u_i, by using Proposition 5 we can show that $y \in H'_k \left(0^+ H'_k\right)^*$. We have $|Q_i| \leq p \leq 2^{k/2} - 1$. So by applying Proposition 22 to the DFA D_i, the states $\delta_i(s_i, f_n v_{i-1} 0)$ and $\delta_i(s_i, g_n v_{i-1} 0)$, and the string y, we obtain that $y' \in H_k \left(0^+ H_k\right)^*$ exists such that

$$\delta_i(s_i, f_n v_{i-1} 0 y) = \delta_i(s_i, f_n v_{i-1} 0 y') \tag{5}$$

and

$$\delta_i(s_i, g_n v_{i-1} 0 y) = \delta_i(s_i, g_n v_{i-1} 0 y'). \tag{6}$$

Now we set

$$v_i = v_{i-1} 0 y'.$$

By Condition 2 for $e = i-1$, we have $v_{i-1} \in z_k \left(0^+ H_k\right)^*$. Thus we obtain

$$v_i \in z_k \left(0^+ H_k\right)^* 0^+ H_k \left(0^+ H_k\right)^* = z_k \left(0^+ H_k\right)^+ \subseteq z_k \left(0^+ H_k\right)^* ,$$

and therefore Condition 2 is satisfied for $e = i$. Afterwards, we set $w_i \coloneqq u_i f_n v_i$ and $x_i \coloneqq u_i g_n v_i$. This satisfies Condition 3 for $e = i$.

By substituting $u_i = v_{i-1}0y$ and $v_i = v_{i-1}0y'$ in equations 5 and 6, we get $\delta_i(s_i, f_n u_i) = \delta_i(s_i, f_n v_i)$ and $\delta_i(s_i, g_n u_i) = \delta_i(s_i, g_n v_i)$. Now consider an arbitrary string $\alpha \in \Sigma^*$. Suppose $\delta_i(\alpha u_i) = s_i$. Hence we get

$$\delta_i(\alpha u_i f_n u_i) = \delta_i(s_i, f_n u_i) = \delta_i(s_i, f_n v_i) = \delta_i(\alpha u_i f_n v_i), \tag{7}$$

and similarly, we have

$$\delta_i(\alpha u_i g_n u_i) = \delta_i(s_i, g_n u_i) = \delta_i(s_i, g_n v_i) = \delta_i(\alpha u_i g_n v_i). \tag{8}$$

Besides, $u_i = C_n(v_{i-1}0u_{i-1})$. So by Lemma 9, we get that $\text{sep}(u_i f_n u_i, u_i g_n u_i)$ is at least $2n+2$. Hence by Lemma 4, we get

$$\text{sep}(\alpha u_i f_n u_i, \alpha u_i g_n u_i) \geq \text{sep}(u_i f_n u_i, u_i g_n u_i) \geq 2n+2.$$

Since $p \leq 2n+1$, no D in M_p can separate $\alpha u_i f_n u_i$ and $\alpha u_i g_n u_i$. Hence by Lemma 3, D_i cannot distinguish $\alpha u_i f_n u_i$ and $\alpha u_i g_n u_i$, so we have

$$\delta_i(\alpha u_i f_n u_i) = \delta_i(\alpha u_i g_n u_i). \tag{9}$$

By equations 7, 8, and 9, we can conclude that if $\delta_i(\alpha u_i) = s_i$, then we have

$$\delta_i(\alpha u_i f_n v_i) = \delta_i(\alpha u_i f_n u_i) = \delta_i(\alpha u_i g_n u_i) = \delta_i(\alpha u_i g_n v_i),$$

or equivalently, by substituting $w_i = u_i f_n v_i$ and $x_i = u_i g_n v_i$, we can write $\delta_i(\alpha w_i) = \delta_i(\alpha x_i)$. Furthermore, it follows that for any string $\alpha' \in \Sigma^*$, we have

$$\delta_i(\alpha w_i \alpha') = \delta_i(\alpha x_i \alpha').$$

Thus Condition 4 is satisfied when $e = j = i$. Moreover, we have $u_i \in \Sigma^* u_{i-1}$ and $v_i \in v_{i-1}\Sigma^*$. So there exist $b, b' \in \Sigma^*$ such that $u_i = bu_{i-1}$ and $v_i = v_{i-1}b'$. Hence we can write

$$w_i = u_i f_n v_i = bu_{i-1} f_n v_{i-1} b' = bw_{i-1}b',$$

and similarly, we have

$$x_i = u_i g_n v_i = bu_{i-1} g_n v_{i-1} b' = bx_{i-1}b'.$$

Thus for all $\alpha, \alpha' \in \Sigma^*$, we have

$$\alpha w_i \alpha' = \alpha b w_{i-1} b' \alpha', \alpha x_i \alpha' = \alpha b x_{i-1} b' \alpha'. \tag{10}$$

Suppose $1 \leq j \leq i-1$. By Condition 4 for $e = i-1$, if

$$\delta_j(\alpha b u_{i-1}) = \delta_j(\alpha u_i) = s_j,$$

then $\delta_j(\alpha b w_{i-1} b' \alpha') = \delta_j(\alpha b x_{i-1} b' \alpha')$, or equivalently, by using equation 10 we can write $\delta_j(\alpha w_i \alpha') = \delta_j(\alpha x_i \alpha')$. So Condition 4 for $e = i$ is also satisfied when $j \leq i-1$. Therefore Condition 4 holds for $e = i$. Hence we proved that all four conditions are satisfied for $e = i$.

In the end, we set $u = u_m, v = v_m, w = w_m$ and $x = x_m$. We claim $\operatorname{sep}(w, x) \geq p+1$. Otherwise, suppose $D \in M_p$ separates w, x. So D distinguishes w and x. Let us write $s = \delta_D(u)$. We have $|Q_D| \leq p$. Therefore by definition, there exists $1 \leq i \leq m$ such that $D_i = D$ and $s_i = s$. Since $\delta_D(u) = s$, by Condition 4 for $e = m$ and $j = i$, we have $\delta_D(w) = \delta_D(x)$, which contradicts the assumption that D separates w and x. Therefore $\operatorname{sep}(w, x) \geq p+1$. Now we set $w' = \operatorname{tr}^R(w)$ and $x' = \operatorname{tr}^R(x)$. By Lemma 24, we have

$$\operatorname{sep}(w', x') \geq \operatorname{sep}(w, x) \geq p + 1 = \min(2n+2, 2^{k/2}).$$

Besides, we have $w^R = v^R f_n u^R$ and $x^R = v^R g_n u^R$. By Condition 2, we have $v \in z_k \left(0^+ H_k\right)^*$. Thus

$$\begin{aligned} v^R \in \left(H_k^R 0^+\right)^* z_k^R &= \left(\left(\{1,2\}^* - G_k^R\right) 0^+\right)^* z_k^R \\ &\subseteq \left(\left(\{1,2\}^* - G_k^R\right) 0^+\right)^* \left(G_k^R - \{\epsilon\}\right). \end{aligned}$$

By Lemma 14, we have $\operatorname{sc}(G_k^R) \leq 5k+3$. Moreover, by Condition 1 we obtain $u^R \in \{1,2\} \Sigma^*$. Therefore by definition, we obtain $\operatorname{tr}(u^R) \in 1\{0,1\}^*$. Hence by Lemma 25, we get

$$\begin{aligned} \operatorname{sep}((w')^R, (x')^R) &= \operatorname{sep}(\operatorname{tr}(w^R), \operatorname{tr}(x^R)) \\ &= \operatorname{sep}(\operatorname{tr}(v^R f_n u^R), \operatorname{tr}(v^R g_n u^R)) \\ &= \operatorname{sep}(\operatorname{tr}(v^R) f_n \operatorname{tr}(u^R), \operatorname{tr}(v^R) g_n \operatorname{tr}(u^R)) \quad &\leq 2(5k+3) + n + 4 \\ & &= 10k + n + 10. \end{aligned}$$

□

Theorem 27. *The difference*

$$\left|\operatorname{sep}(w, x) - \operatorname{sep}(w^R, x^R)\right|$$

is unbounded for an alphabet of size at least 2.

Proof. Let k be a positive even integer. If we set $n = 2^{k/2-1} - 1$, then by Theorem 26, there exist strings w and x in $\{0,1\}^*$ such that

$$\operatorname{sep}(w,x) \geq \min(2n+2, 2^{k/2}) = 2^{k/2},$$

and

$$\operatorname{sep}(w^R, x^R) \leq n + 10k + 10 = (2^{k/2-1} - 1) + 10k + 10.$$

So we have

$$\begin{aligned} \operatorname{sep}(w,x) - \operatorname{sep}(w^R, x^R) &\geq 2^{k/2} - \left(2^{k/2-1} + 10k + 9\right) \\ &= 2^{k/2-1} - 10k - 9, \end{aligned}$$

which tends to infinity as k approaches infinity. □

3. Conclusion

In this paper, we proved that the difference $\left|\operatorname{sep}(w,x) - \operatorname{sep}(w^R, x^R)\right|$ can be unbounded. However, it remains open to determine whether there is a good upper bound on $\operatorname{sep}(w,x)/\operatorname{sep}(w^R, x^R)$.

Acknowledgments

I wish to thank Jeffrey Shallit, Mohammad Izadi, Arseny Shur, MohammadTaghi Hajiaghayi, Keivan Alizadeh, Hooman Hashemi, Hadi Khodabandeh, and Mobin Yahyazadeh, who helped me write this paper. I would also like to thank the anonymous referees for their careful reading of this paper, and for their valuable comments and suggestions.

[1] Pavel Goralčík and Václav Koubek. On discerning words by automata. In *Automata, Languages and Programming, 13th International Colloquium, ICALP86, Rennes, France, July 15-19, 1986, Proceedings*, pages 116–122, 1986.

[2] Erik D. Demaine, Sarah Eisenstat, Jeffrey Shallit, and David A. Wilson. Remarks on separating words. In *Descriptional Complexity of Formal Systems - 13th International Workshop, DCFS 2011, Gießen/Limburg, Germany, July 25-27, 2011. Proceedings. LNCS, vol. 6808*, pages 147–157, 2011.

[3] John M. Robson. Separating strings with small automata. *Inf. Process. Lett.*, 30(4):209–214, 1989.

[4] John M. Robson. Separating words with machines and groups. *ITA*, 30 (1):81–86, 1996.

[5] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to automata theory, languages, and computation - (2. ed.).* Addison-Wesley series in computer science. Addison-Wesley-Longman, 2001. ISBN 978-0-201-44124-6.

[6] Yuan Gao, Lila Kari, and Sheng Yu. State complexity of union and intersection of square and reversal on k regular languages. *Theor. Comput. Sci.*, 454:164–171, 2012.

[7] Galina Jirásková. On the state complexity of complements, stars, and reversals of regular languages. In *Developments in Language Theory, 12th International Conference, DLT 2008, Kyoto, Japan, September 16-19, 2008. Proceedings*, pages 431–442, 2008.

[8] Juraj Šebej. Reversal of regular languages and state complexity. In *Proceedings of the Conference on Theory and Practice of Information Technologies, ITAT 2010, Hotel Smrekovica, Veľká Fatra, Slovak Republic, September 21-25, 2010*, pages 47–54, 2010.

[9] A. N. Maslov. Estimates of the number of states of finite automata. In *Soviet Mathematics Doklady*, volume 11, pages 1373–1375, 1970.

[10] Sheng Yu, Qingyu Zhuang, and Kai Salomaa. Obtaining tight upper bounds for the state complexities of DFA operations. In *Computing and Information - ICCI'92, Fourth International Conference on Computing and Information, Toronto, Ontario, Canada, May 28-30, 1992, Proceedings*, pages 100–104, 1992.

[11] Sheng Yu, Qingyu Zhuang, and Kai Salomaa. The state complexities of some basic operations on regular languages. *Theor. Comput. Sci.*, 125 (2):315–328, 1994.