State Complexity of Prefix Distance[☆]

Timothy Ng*, David Rappaport*, Kai Salomaa*

School of Computing, Queen's University, Kingston, Ontario K7L 2N8, Canada

Abstract

The prefix distance between strings x and y is the number of symbol occurrences in the strings that do not belong to the longest common prefix of x and y. The suffix and the substring distances are defined analogously in terms of the longest common suffix and longest common substring, respectively, of two strings. We show that the set of strings within prefix distance k from an n state DFA (deterministic finite automaton) language can be recognized by a DFA with $(k+1) \cdot n - \frac{k(k+1)}{2}$ states and that this number of states is needed in the worst case. Also we give tight bounds for the nondeterministic state complexity of the set of strings within prefix, suffix or substring distance k from a regular language.

1. Introduction

Various similarity measures between strings and languages have been considered for information transmission applications. The edit distance counts the number of substitution, insertion and deletion operations that are needed to transform one string to another. The Hamming distance counts the number of positions in which two equal length strings differ. A distance measure between words can be extended in various ways as a distance between sets of strings (or languages) [3, 4] and algorithms for computing the distance between languages are important for error-detection and error-correction applications [4, 9, 10]. The descriptional complexity of error/edit systems has been considered by Kari and Konstantinidis [8]. Other types of sequence similarity measures have been considered e.g. by Apostolico [1].

Instead of counting the number of edit operations, the similarity of strings can be defined by way of their longest common prefix, suffix, or substring, respectively [4]. For example, the prefix distance of strings x and y is the sum

[☆]An extended abstract of this paper appeared in the Proceedings of the 20th International Conference Implementation and Application of Automata, Umeå, Sweden, August 18–21, 2015.

*Corresponding author.

Email addresses: ng@cs.queensu.ca (Timothy Ng), daver@cs.queensu.ca (David Rappaport), ksalomaa@cs.queensu.ca (Kai Salomaa)

of the length of the suffix of x and the suffix of y that occurs after their longest common prefix. A parameterized prefix distance between regular languages has been considered by Kutrib et al. [11] for estimating the fault tolerance of information transmission applications.

The neighbourhood of radius k of a language L consists of all strings that are within distance k from some string in L. Calude et al. [3] have shown that the neighbourhood of a regular language with respect to an additive distance is regular. A distance is said to be additive if it, in a certain sense, respects string concatenation. This gives rise to the question how large is the (non)deterministic finite automaton (DFA, respectively, NFA) needed to recognize the neighbourhood of a regular language, that is, what is the state complexity of neighbourhoods of regular languages.

Povarov [16] has given an improved upper bound and a closely matching lower bound for the state complexity of Hamming neighbourhoods of radius one. Upper bounds for the state complexity of neighbourhoods with respect to an additive distance or quasi-distance have been obtained by the authors [14, 17] using a construction based on weighted finite automata and a matching lower bound was given recently in [15].

It follows from Choffrut and Pighizzini [4] that the prefix, suffix and substring distances preserve regularity, that is, the neighbourhood of a regular language of finite radius remains regular. Here we study the state complexity of these neighbourhoods. The neighbourhood of radius r of a language L with respect to the prefix distance, roughly speaking, consists of strings that share a "long" prefix with a string $u \in L$, more precisely, it is required that the combined length of the parts of w and u outside their longest common suffix is at most the constant r. In view of this it seems reasonable to expect that the state complexity of prefix distance neighbourhoods does not incur a similar exponential size blow-up as the edit distance [15].

We show that if L is recognized by a deterministic finite automaton (DFA) of size n, the prefix neighbourhood of L of radius $k < n$ has a DFA of size $(k+1) \cdot n - \frac{k(k+1)}{2}$ and that this bound cannot be improved in the worst case. Our lower bound construction uses an alphabet of size $n-1$ and we show that the general upper bound cannot be reached using languages defined over a fixed alphabet.

We consider also the nondeterministic state complexity of prefix, suffix and substring neighbourhoods. If L has a nondeterministic finite automaton (NFA) of size n, the neighbourhood of L of radius k can be recognized by an NFA of size $n+k$. The upper bound for the substring neighbourhood of L of radius k is $(k+1) \cdot n + 2k$. In all cases we give matching lower bounds for nondeterministic state complexity, and in the lower bound constructions L has, in fact, a DFA of size n.

2. Preliminaries

Here we briefly recall some definitions and notation used in the paper. For all unexplained notions on finite automata and regular languages the reader

may consult the textbook by Shallit [18] or the survey by Yu [19]. A survey of distances is given by Deza and Deza [5]. Recent surveys on descriptional complexity of regular languages include [6, 7, 12].

In the following Σ is always a finite alphabet, the set of strings over Σ is Σ^* and ε is the empty string. The reversal of a string $x \in \Sigma^*$ is x^R. The set of nonnegative integers is $\mathbb{N}_0$. The cardinality of a finite set S is denoted $|S|$ and the powerset of S is 2^S. A string $w \in \Sigma^*$ is a *substring* or *factor* of x if there exist strings $u, v \in \Sigma^*$ such that $x = uwv$. If $u = \varepsilon$, then w is a *prefix* of x. If $v = \varepsilon$, then w is a *suffix* of x.

A *nondeterministic finite automaton* (NFA) is a 5-tuple $A = (Q, \Sigma, \delta, Q_0, F)$ where Q is a finite set of states, Σ is an alphabet, δ is a multi-valued transition function $\delta : Q \times \Sigma \to 2^Q$, $Q_0 \subseteq Q$ is a set of initial states, and $F \subseteq Q$ is a set of final states. We extend the transition function δ to $Q \times \Sigma^* \to 2^Q$ in the usual way. A string $w \in \Sigma^*$ is *accepted* by A if, for some $q_0 \in Q_0$, $\delta(q_0, w) \cap F \neq \emptyset$ and the language recognized by A consists of all strings accepted by A. An ε-NFA is an extension of an NFA where transitions can be labeled by the empty string ε [18, 19], i.e., δ is a function $Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$. It is known that every ε-NFA has an equivalent NFA without ε-transitions and with the same number of states. An NFA $A = (Q, \Sigma, \delta, Q_0, F)$ is a *deterministic finite automaton* (DFA) if $|Q_0| = 1$ and, for all $q \in Q$ and $a \in \Sigma$, $\delta(q, a)$ either consists of one state or is undefined. Two states p and q of a DFA A are equivalent if $\delta(p, w) \in F$ if and only if $\delta(q, w) \in F$ for every string $w \in \Sigma^*$. A DFA A is *minimal* if each state $q \in Q$ is reachable from the initial state and no two states are equivalent.

Note that our definition of a DFA allows some transitions to be undefined, that is, by a DFA we mean an incomplete DFA. It is well known that, for a regular language L, the sizes of the minimal incomplete and complete DFAs differ by at most one. The constructions in Section 3 are more convenient to formulate using incomplete DFAs but our results would not change in any significant way if we were to require that all DFAs are complete.

The (incomplete deterministic) *state complexity* of a regular language L, $\mathrm{sc}(L)$, is the size of the minimal DFA recognizing L. The *nondeterministic state complexity* of L, $\mathrm{nsc}(L)$, is the size of a minimal NFA recognizing L. A minimal NFA recognizing a regular language need not be unique. A common way of establishing lower bounds for nondeterministic state complexity relies on fooling sets.

Definition 1. A set of pairs of strings $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, $x_i, y_i \in \Sigma^*$, $i = 1, \ldots, m$, is a *fooling set* for a language L if $x_i y_i \in L$, $i = 1, \ldots, m$ and, for all $1 \leq i < j \leq m$, $x_i y_j \notin L$ or $x_j y_i \notin L$.

Proposition 1 ([2, 7])**.** *If L has a fooling set S then* $\mathrm{nsc}(L) \geq |S|$.

To conclude this section, we recall definitions of the distance measures used in the following. Generally, a function $d : \Sigma^* \times \Sigma^* \to [0, \infty)$ is a *distance* if it satisfies for all $x, y, z \in \Sigma^*$, the conditions $d(x, y) = 0$ if and only if $x = y$, $d(x, y) = d(y, x)$, and $d(x, z) \leq d(x, y) + d(y, z)$. The *neighbourhood* of a

language L of radius k with respect to a distance d is the set

$$E(L, d, k) = \{w \in \Sigma^* \mid (\exists x \in L) d(w, x) \leq k\}.$$

Let $x, y \in \Sigma^*$. The *prefix distance* of x and y counts the number of symbols which do not belong to the longest common prefix of x and y [4]. It is defined by

$$d_p(x, y) = |x| + |y| - 2 \cdot \max_{z \in \Sigma^*} \{|z| \mid x, y \in z\Sigma^*\}.$$

Similarly, the *suffix distance* of x and y counts the number of symbols which do not belong to the longest common suffix of x and y and is defined

$$d_s(x, y) = |x| + |y| - 2 \cdot \max_{z \in \Sigma^*} \{|z| \mid x, y \in \Sigma^* z\}.$$

The *substring distance* measures the similarity of x and y based on their longest common continuous substring (or factor) and is defined

$$d_f(x, y) = |x| + |y| - 2 \cdot \max_{z \in \Sigma^*} \{|z| \mid x, y \in \Sigma^* z \Sigma^*\}.$$

The paper [4] refers to d_f as the *subword distance.* The term "subword distance" has been used also for a distance defined in terms of the longest common noncontinuous subword [13].

3. State Complexity of Prefix Neighbourhoods

In this section we consider the deterministic state complexity of prefix neighbourhoods. We construct a DFA for the neighbourhood of radius k with respect to the prefix distance d_p. After that we show that the construction is optimal by giving a matching lower bound. The lower bound construction uses an alphabet of size $n+1$ where n is the number of states of the DFA. We show that the upper bound cannot be reached by languages defined over a constant size alphabet.

First, we define a function for use in the following proofs. For a given NFA $A = (Q, \Sigma, \delta, q_0, F)$, we define the function $\varphi_A : Q \to \mathbb{N}_0$ by

$$\varphi_A(q) = \min_{w \in \Sigma^*} \{|w| \mid \delta(q, w) \in F\}. \tag{1}$$

The function $\varphi_A(q)$ gives the length of the shortest path from the state q to a reachable final state. Note that under this definition, if $q \in F$, then $\varphi_A(q) = 0$.

Proposition 2. *Let $n > k \geq 0$ and L be a regular language recognized by a DFA with n states. Then there is a DFA recognizing $E(L, d_p, k)$ with at most $n \cdot (k+1) - \frac{k(k+1)}{2}$ states.*

Proof. Let $A = (Q, \Sigma, \delta, q_0, F)$ be the DFA that recognizes L. We construct a DFA $A' = (Q', \Sigma, \delta', q_0', F')$ that recognizes the neighbourhood $E(L, d_p, k)$. We define the state set

$$Q' = ((Q - F) \times \{1, \ldots, k+1\}) \cup F \cup \{p_1, \ldots, p_k\}.$$

The machine A' has three types of states.

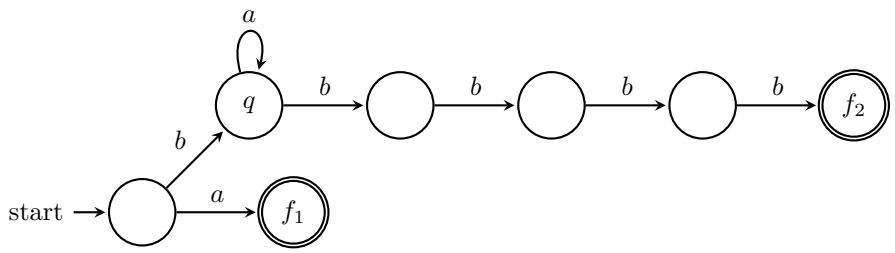

Figure 1: The final state f_1 is closer to q than f_2 even though f_1 is not reachable from q.

- States $q_f \in F$ are final states of A. A word that reaches q_f is a word in $L(A)$.
- States p_ℓ, $1 \leq \ell \leq k$, are reached from the other types of states only on a transition that was undefined in A. A state p_ℓ can be reached from $p_{\ell-1}$ on any symbol $a \in \Sigma$ and can only reach states $p_{\ell'}$ where $\ell \leq \ell' \leq k$. A word that reaches p_ℓ is not a prefix of a word in $L(A)$ and has a distance of ℓ from $L(A)$.
- States $(i, j) \in (Q - F) \times \{1, \ldots, k + 1\}$ are non-final states of A with a counter component. If a word w reaches a state (i, j) in A', then it is prefix of a word recognized by A and is j steps away from the closest final state of A. Note that the closest final state does not need to be reachable from the state i. The closest final state would have been reachable earlier in the computation of w and may not necessarily be reachable from i.

For example, in Figure 1, the word b reaches the state q and is two steps away from the final state f_1 and is four steps away from the final state f_2. In this case, f_1 is considered the closest final state to q even though f_1 is not reachable from q. However, the word baa also reaches q but is three steps away from f_1 and is still four steps away from f_2.

Note that some states of Q' are always unreachable and at the end of the proof we calculate an upper bound for the number of reachable states.

The initial state q'_0 is defined

$$q'_0 = \begin{cases} q_0, & \text{if } q_0 \in F; \\ (q_0, \varphi_A(q_0)) & \text{if } q_0 \notin F \text{ and } \varphi_A(q_0) \leq k; \\ (q_0, k+1) & \text{if } q_0 \notin F \text{ and } \varphi_A(q_0) > k. \end{cases}$$

The set of final states is given by

$$F' = ((Q - F) \times \{1, \ldots, k\}) \cup F \cup \{p_1, \ldots, p_k\}.$$

Let $q_{i,a} = \delta(i, a)$ for $i \in Q$ and $a \in \Sigma$, if $\delta(i, a)$ is defined. Then for all $a \in \Sigma$,

the transition function δ' is defined for states $i \in F$ by

$$\delta'(i,a) = \begin{cases} (q_{i,a}, 1), & \text{if } q_{i,a} \in Q - F; \\ q_{i,a}, & \text{if } q_{i,a} \in F; \\ p_1, & \text{if } \delta(i,a) \text{ is undefined.} \end{cases}$$

For states $(i,j) \in (Q-F) \times \{1, \ldots, k+1\}$, δ' is defined

$$\delta'((i,j),a) = \begin{cases} q_{i,a}, & \text{if } q_{i,a} \in F; \\ (q_{i,a}, \min\{j+1, \varphi_A(q_{i,a})\}), & \text{if } \varphi_A(q_{i,a}) \text{ or } j+1 \leq k; \\ (q_{i,a}, k+1), & \text{if } \varphi_A(q_{i,a}) \text{ and } j+1 > k; \\ p_{j+1}, & \text{if } \delta(i,a) \text{ is undefined.} \end{cases}$$

Finally, we define δ' for states p_ℓ for $\ell = 1, \ldots, k-1$ by

$$\delta'(p_\ell, a) = p_{\ell+1}.$$

We now show that a word $w \in \Sigma^*$ is recognized by A' if and only if w is in the neighbourhood $E(L(A), d_p, k)$. Let $x \in L$ be a closest string to w according to the prefix distance d_p. There are three cases to consider.

1. First, suppose that $x = wx'$ for some $x' \in \Sigma^*$. Then $w \in E(L, d_p, k)$ if and only if $|x'| \leq k$. Consider the computation on w. Since w is a proper prefix of a word in L, the computation cannot end in a state p_ℓ. if the computation ends in a final state in F, then $x = w$ and w is in both $E(L, d_p, k)$ and $L(A')$. Now suppose the computation ends in a state (i,j). Since x is the closest word in L to w, there must be a shortest path of length $|x'|$ in the original DFA A from state i to a final state of A. By definition, (i,j) is a final state if $j = \varphi_A(i) \leq k$. Thus, $j = \varphi_A(i) = |x'|$ and (i,j) is a final state if and only if $j = |x'| \leq k$.
2. Next, suppose that $w = xw'$ for some $w' \in \Sigma^+$. In this case, $w \in E(L, d_p, k)$ if and only if $|w'| \leq k$. The machine reaches some final state f of A once it reads all of x. Then the machine continues reading w' until it reaches some state $q \in Q'$. The state q is either a state (i,j) or a state p_ℓ, since otherwise, $q \in F$ and $w' = \varepsilon$.
   (a) Consider $q = (i,j)$. By definition, (i,j) is a final state if $j \leq k$. Since x is a closest word in L to w, $j = |w'|$ must be the distance of the current computation from the closest final state f unless $|w'| > k$, in which case $j = k+1$. Otherwise, if $j < |w'|$, there was some state (i', j') that was encountered during the computation of w' with a final state f' that was closer than f. Thus, if $|w'| > k$, then $j = k+1$ and (i,j) is not a final state. Otherwise, $j = |w'| \leq k$ and (i,j) is a final state.
   (b) Now consider when $q \neq (i,j)$ and let $w' = w'_1 w'_2$. The computation from f on w'_1 reaches some state $q' = (i', j')$ for which there is no transition in A defined for the first symbol of w'_2. By the same

reasoning as above, $j' = |w'_1| < k$. Since an undefined transition was encountered on the first symbol of w'_2, the machine goes to state $p_{|w'_1|+1}$. From state $p_{|w'_1|+1}$, the machine reads the rest of w'_2. Now, if $|w'| > k$, then $|w'_2| > k - |w'_1|$ and the computation on the rest of w'_2 fails when it reaches p_k and there are no further transitions. Otherwise, $|w'| \leq k$ and the computation of w'_2 ends in a state $p_{|w'_1|+|w'_2|}$, which is a final state since $|w'| = |w'_1| + |w'_2| \leq k$.

3. Finally, suppose that $w = pw'$ and $x = px'$ with $p \in \Sigma^*$ and $w', x' \in \Sigma^+$ such that p is the longest common prefix of w and x. Thus $w \in E(L, d_p, k)$ if and only if $|w'| + |x'| \leq k$. In this case, A' reads w until it reaches a state (i_p, j_p) on the prefix p. At this point, reading x' from (i_p, j_p) will take the machine to some final state $f \in F$, while reading w' from (i_p, j_p) takes the machine to some other state $q \in Q'$. Note that $|x'| \leq k$, since otherwise $|x'| + |w'| > k$, and $j_p = \varphi_A(i_p) = |x'|$, since otherwise x would not be a closest word to w. Now, q is either of the form (i, j) or a state p_ℓ.
   (a) Suppose q is of the form (i, j). Then j is either $|w'| + |x'|$ or $k + 1$. If $|w'| > k - |x'|$, then $j = k + 1$ and (i, j) is not a final state. If $j \leq |w'| + |x'|$, then there must be some final state $f' \in F$ closer to a state on the computation path of w' from (i_p, j_p) which cannot be the case if x is a closest word to w. Thus, $j = |w'| + |x'| \leq k$ and (i, j) is a final state.
   (b) Now, suppose $q \neq (i, j)$ and let $w' = w'_1 w'_2$. The computation from (i_p, j_p) on w'_1 reaches some state $q' = (i', j')$ for which there is no transition in A on the first symbol of w'_2. By the same reasoning as above, $j' = |w'_1| < k - |x'|$. Since an undefined transition was encountered on the first symbol of w'_2, the machine goes to state $p_{|w'_1|+|x'|+1}$. From state $p_{|w'_1|+|x'|+1}$, the rest of w'_2 is read. If $|w'_2| > k-(|w'_1|+|x'|)$, then the computation of w'_2 falls off at p_k. Otherwise, the computation ends in state $p_{|w'_2|+|w'_1|+|x'|}$. We have

$$|w'_2| + |w'_1| + |x'| = |w'| + |x'| \leq k$$

   and thus, $p_{|w'_2|+|w'_1|+|x'|}$ is a final state.

If $f = |F|$, then the set of states Q' has $(n - f) \cdot (k + 1) + k + f$ elements but they cannot all be reachable. Based on the definition of the transitions of δ' we observe that if there is a transition entering a state (q, j), $q \in Q - F$, $1 \leq j \leq k + 1$, then $\varphi_A(q)$ must be at least j. Thus, all elements of the set

$$S_{ur} = \{(q, j) \mid q \in Q - F, 1 \leq j \leq k + 1,\ j > \varphi_A(q)\}$$

are unreachable as states of A'.

First we consider the case where A has at least k non-final states, that is, $|Q - F| \geq k$. Since all states of A are useful, each non-final state must have a path leading to a final state. Now the cardinality of S_{ur} is minimized, informally speaking, when there are as few as possible non-final states q for

which $\varphi_A(q)$ is "small". By the definition of the function φ_A, if for some state p the value $\varphi_A(p) = \ell \geq 2$, then there must exist another non-final state p' with $\varphi_A(p) = \ell - 1$. This means that the cardinality of S_{ur} is minimized when in the DFA A for each $1 \leq i \leq k$, exactly one non-final state q_i has a shortest path of length i that reaches a final state, and for all other non-final states the length of the shortest path to a final state is greater than k. In this case, $S_{ur} = \{(q_i, j) \mid i < j \leq k+1,\ i = 1, \ldots, k\}$ and $|S_{ur}| = \frac{k(k+1)}{2}$ and the total number of reachable states of A' is at most

$$(n - f) \cdot (k + 1) + k + f - \frac{k(k+1)}{2}.$$

This value is maximized as $n \cdot (k+1) - \frac{k(k+1)}{2}$ by choosing $f = 1$.

Finally, it remains to consider the case where $|Q - F| < k$, that is, $f > n - k$. With this assumption we get

$$|Q'| = n \cdot (k+1) - (f-1) \cdot k \leq n \cdot (k+1) - (n-k-1) \cdot k = n - k \cdot (k+1).$$

Thus, when A has fewer than k non-final states the size of A' is less than the claimed upper bound even assuming that all states of Q' were reachable.

We have verified that at most $n \cdot (k+1) - \frac{k(k+1)}{2}$ states of A' can be reachable. □

The lower bound construction that we present uses an alphabet with variable size. We will show later that it is impossible to reach the upper bound (for all n) with an alphabet of fixed size.

Lemma 3. *For $n > k \in \mathbb{N}$, there exists a DFA A_n with n states over an alphabet of size $n-1$ such that*

$$\mathrm{sc}(E(L(A_n), d_p, k)) \geq n \cdot (k+1) - \frac{k(k+1)}{2}.$$

Proof. We define a DFA $A_n = (Q_n, \Sigma_n, \delta_n, q_0, F)$ (Figure 2) by choosing

$$Q_n = \{0, \ldots, n-1\}, \quad \Sigma_n = \{a, b, c_2, \ldots, c_{n-2}\},$$

$q_0 = 0$, $F = \{0\}$, and the transition function is given by

- $\delta_n(0, a) = 1$
- $\delta_n(q, a) = q$ for $q = 1, \ldots, n-1$,
- $\delta_n(q, b) = q + 1 \bmod n$ for $q = 1, \ldots, n-1$,
- $\delta_n(0, c_i) = i$ for $i = 2, \ldots, n-2$,

Note that for every state $q \in Q_n - \{0\}$, we have $\varphi_{A_n}(q) = n - q$.

We transform A_n into the DFA $A'_n = (Q'_n, \Sigma_n, \delta'_n, q'_0, F')$ by following the construction from Proposition 2. To determine the reachable states of Q'_n, we

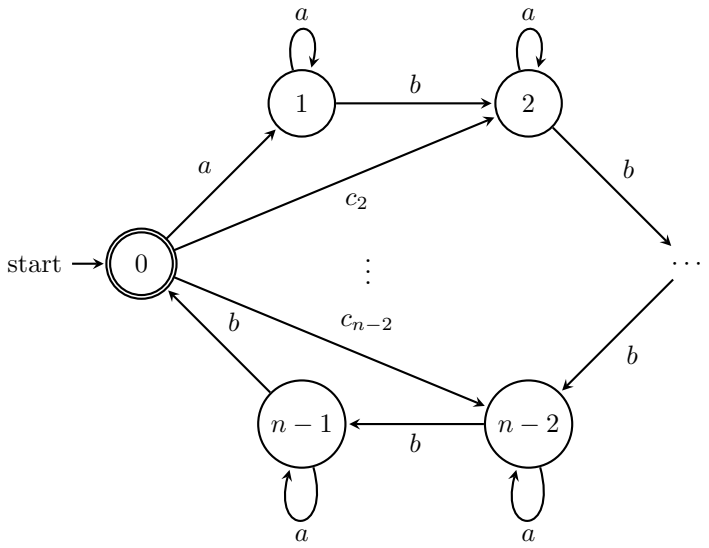

Figure 2: The DFA A_n.

first consider states of the form $(i, j) \in (Q_n - \{0\}) \times \{1, \ldots, k+1\}$. For state 1, we can reach state $(1, j)$ via the word $a \cdot a^j$. For states $i \in \{2, \ldots, n-k-1\}$ with $\varphi_{A_n}(i) > k$, we can reach state (i, j) via the word $c_i a^j$ for $j = 1, \ldots, k+1$. For states $i \in \{n-k, \ldots, n-2\}$ with $\varphi_{A_n}(i) \leq k$, we can reach state (i, j) via the word $c_i a^j$ for $j = 1, \ldots, \varphi_{A_n}(i)$. However, states (i, j) with $j > \varphi_{A_n}(i)$ are unreachable by definition of A'_n. For state $n-1$, we can reach $(n-1, 1)$ via the word ab^{n-2} and states $(n-1, j)$ are unreachable for $j > 1$ since $\varphi_{A_n}(n-1) = 1$. Thus the number of unreachable states in $(Q_n - \{0\}) \times \{1, \ldots, k+1\}$ is

$$\sum_{i=n-k}^{n-1} |\{i\} \times \{\varphi_{A_n}(i)+1, \ldots, k+1\}| = \sum_{i=1}^{k} |\{i+1, \ldots, k+1\}| = \sum_{i=1}^{k} i = \frac{k(k+1)}{2}.$$

Now consider states $p_1, \ldots, p_k$. The state p_ℓ is reachable on the word b^ℓ. Finally, 0 is reachable since it is the initial state. Thus, the number of reachable states is

$$(n-1) \cdot (k+1) - \frac{k(k+1)}{2} + k + 1 = n \cdot (k+1) - \frac{k(k+1)}{2}.$$

Now, we show that all reachable states are pairwise inequivalent. First, note that 0 can be distinguished from any other state by the word ε and that as a final state of A, 0 is not a state of the form (i, j) in A'. Next, we distinguish states of the form (i, j) from states of the form p_ℓ via the word $a^k b^{n-i}$. From state (i, j), reading a^k takes the machine to state $(i, \min\{\varphi_{A_n}(i), k+1\})$. Subsequently reading b^{n-i} takes the machine to the final state 0. However, for every state p_ℓ, reading a^k forces the machine beyond state p_k, after which there are no transitions defined.

Next, without loss of generality, we let $\ell < \ell'$ and consider states p_ℓ and $p_{\ell'}$. From above, the state p_ℓ can be reached by a word b^ℓ and $p_{\ell'}$ is reached by a word $b^{\ell'}$. Choose $z = a^{k-\ell}$. The string z takes state p_ℓ to the state p_k, where it is accepted. However, the computation on string z from state $p_{\ell'}$ is undefined since $\ell' + k - \ell > k$.

Finally, we consider states of the form (i, j). Let $i < i'$ and consider states (i, j) and (i', j'). Let $z = b^{n-i+k}$. From state (i, j), the word z goes to state 0 on b^{n-i}. Then by reading b^k from state 0, we reach state p_k, an accepting state. However, when reading z from state (i', j'), we reach state 0 on $b^{n-i'}$, since $i' > i$. We are then left with $b^{i'-i+k}$. Reading b^k takes us to state p_k, where we still have $b^{i'-i}$ to read and no further defined transitions. Thus, the computation is rejected.

Next, we fix i and let $j < j'$. First, consider the case when $\varphi_{A_n}(i) > k$. Then let $z = a^{k-j}$. Reading z from (i, j) takes us to state (i, k), which is a final state. However, from (i, j'), reading z brings us to state $(i, k + 1)$ and so the computation is rejected.

Now, consider the case when $\varphi_{A_n}(i) \leq k$. Let $z = c_{n-2}a^{k-j-1}$. From state (i, j), reading c_{n-2} takes the machine to state p_{j+1} and reading a^{k-j-1} puts the machine in the accepting state p_k. However, reading z from (i, j') takes us to state p_k with $a^{j'-j}$ still unread since $j' + k - j - 1 > k$ and thus, with no further transitions available, the computation is rejected.

Thus, we have shown that there are $n \cdot (k + 1) - \frac{k(k+1)}{2}$ reachable states and that all reachable states are pairwise inequivalent. □

Taking Proposition 2 together with Lemma 3, we get the following theorem.

Theorem 4. *For $n > k \geq 0$, if $\mathrm{sc}(L) = n$ then*

$$\mathrm{sc}(E(L, d_p, k)) \leq n \cdot (k + 1) - \frac{k(k + 1)}{2}$$

and this bound can be reached in the worst case.

The proof of Lemma 3 uses an alphabet of size $n-1$. To conclude this section we observe that the general upper bound cannot be reached by languages defined over an alphabet of size less than $n - 1$.

Proposition 5. *Let A be a DFA with n states. If the state complexity of $E(L(A), d_p, k)$ equals $n \cdot (k + 1) - \frac{k(k+1)}{2}$, then the alphabet of A needs at least $n - 1$ letters.*

Proof. Let $A = (Q, \Sigma, \delta, q_0, F)$ with $|Q| = n$. Let $A' = (Q', \Sigma, \delta', q'_0, F')$ be the DFA recognizing $E(L(A), d_p, k)$ constructed in the proof of Proposition 2. In the analysis at the end of the proof it was observed that in order for A' to have the maximal number of states $n \cdot (k+1) - \frac{k(k+1)}{2}$, a necessary condition is that F is a singleton set, $F = \{q_f\}$ and there can be only one state q_1 with $\varphi_A(q_1) = 1$ (that is, q_1 is a non-final state with a direct transition to q_f).

Now for all $r \in Q - \{q_f, q_1\}$, $\varphi_A(r) \geq 2$. Based on the definition of the transitions of δ', if $\varphi_A(r) \geq 2$ the state $(r, 1)$ can be reached only by a (direct)

transition from a final state, that is, from q_f. Thus, in the DFA A the state q_f must have at least $n-2$ outgoing transitions. Furthermore, on some input symbol the transition from q_f must be undefined in order for the error state p_1 to be reachable in A'. (As defined in a construction of A' in the proof of Proposition 2, p_1 can be reached only by a direct transition from a final state.)

Since A is a DFA, q_f has at least $n-2$ outgoing transitions and at least one undefined transition, the cardinality of the alphabet must be at least $n-1$.

□

4. Nondeterministic State Complexity

We consider the nondeterministic state complexity of neighbourhoods of a regular language with respect to the prefix-, the suffix- and the substring distance, respectively.

4.1. Prefix and Suffix Distance

We consider first neighbourhoods with respect to the prefix distance, and the results for the suffix distance are obtained as a consequence of the fact that the nondeterministic state complexity of a regular language L is the same as the nondeterministic state complexity of the reversal of L and using the observation $d_s(x,y) = d_p(x^R, y^R)$ for all strings x and y.

We give an upper bound for the nondeterministic state complexity of the neighbourhood of radius k with respect to the prefix distance d_p and give a matching lower bound construction.

Proposition 6. *Let $k \geq 0$ and L be a regular language recognized by an NFA with n states. Then there is an NFA recognizing $E(L, d_p, k)$ with at most $n+k$ states.*

Proof. Let $A = (Q, \Sigma, \delta, Q_0, F)$ be the NFA recognizing L. We define an NFA $A' = (Q', \Sigma, \delta', I, F)$ for the language $E(L, d_p, k)$ by

- $Q' = Q \cup \{p_1, \ldots, p_k\}$, $I = Q_0$,
- $F' = F \cup \{p_1, \ldots, p_k\} \cup \{q \in Q \mid \varphi_A(q) \leq k\}$.

The state set Q' consists of all the original states in Q and k new states p_i, $1 \leq i \leq k$. Recall from (1) that each state $q \in Q$ has an associated value $\varphi_A(q)$ which denotes the length of the shortest string that takes q to a final state in F. Thus, for any word $u \in \Sigma^*$ that reaches a state $q \in Q$, there is a word $v \in \Sigma^*$ of length $\varphi_A(q)$ such that $uv \in L(A)$. States p_i, $1 \leq i \leq k$, are reached on a transition on any symbol from a state $q \in Q$ with $\varphi_A(q) = i-1$ or from p_{i-1}. We will show later that if a word $w \in \Sigma^*$ has a computation that ends in a state p_i then $d_p(w, L(A)) = i$.

The transition function is defined for all $a \in \Sigma$ by

- $\delta'(q,a) = \delta(q,a) \cup \{p_1\}$ for all $q \in F$,

- $\delta'(q,a) = \delta(q,a) \cup \{p_{\varphi_A(q)+1}\}$ for all $q \in Q$ with $\varphi_A(q) < k$,
- $\delta'(p_i,a) = p_{i+1}$ for $i = 1,\ldots,k-1$.

The transitions of A' are the same as those of A with the addition of transitions to states p_i, $1 \leq i \leq k$. For each state $q \in Q$, we add transitions on every symbol $a \in \Sigma$ to the state $p_{\varphi_A(q)+1}$ if $\varphi_A(q) \leq k$.

We now show that a word $w \in \Sigma^*$ is recognized by A' if and only if w is in the neighbourhood $E(L(A), d_p, k)$. Let $x \in L$ be a closest string to w according to the prefix distance d_p. There are three cases to consider.

First, suppose that $x = wx'$ for some $x' \in \Sigma^*$. Then $w \in E(L, d_p, k)$ if and only if $|x'| \leq k$. The computation of w ends in a state q. By definition, q is a final state if and only if $\varphi_A(q) \leq k$. Since x is a closest string to w in L, the shortest path from q to a final state of A must be of length $|x'|$. Thus, $\varphi_A(q) = |x'|$ and q is a final state if and only if $|x'| \leq k$.

Next, suppose that $w = xw'$ for some $w' \in \Sigma^*$. Then $w \in E(L, d_p, k)$ if and only if $|w'| \leq k$. In this case, during the computation of w, a final state f will be reached once x has been read. If $|w'| \leq k$, then there is at least one computation path which reaches $p_{|w'|}$ when reading w' from f. If $|w'| > k$ then there are no accepting computations. Otherwise, an accepting computation must reach another state q with $\varphi_A(q) \leq k$ and x would no longer be a closest word to w. Thus, A' recognizes w if and only if $|w'| \leq k$.

Finally, suppose that $w = pw'$ and $x = px'$ with $p, w', x' \in \Sigma^*$. Then $w \in E(L, d_p, k)$ if and only if $|w'| + |x'| \leq k$. The NFA A' begins reading w until it reaches a state q once it reads the prefix p. Since x is a closest string to w in L, there must exist a shortest path of length $|x'|$ from q to a final state of A. Thus, $\varphi_A(q) = |x'|$. If $|x'| > k$, then $w \notin E(L, d_p, k)$. Thus, $\varphi_A(q) = |x'| \leq k$ and there exists a computation of w' which transitions to state $p_{|x'|+1}$ and ends in state $p_{|w'|+|x'|}$ if $|w'| \leq k - |x'|$. Now, if $|w'| > k - |x'|$, there are no accepting computations. Otherwise, an accepting computation would reach another state q with $\varphi_A(q) \leq |x'| + |w'|$ and x would no longer be a closest word to w. Thus, A' accepts w if and only if $|w'| < k - \varphi_A(q) = k - |x'|$.

□

Using the fooling sets of Proposition 1 we get a matching lower bound.

Lemma 7. *For $n, k \in \mathbb{N}$, there exists a DFA A with n states over $\Sigma = \{a, b\}$ such that any NFA for $E(L(A), d_p, k)$ requires $n + k$ states.*

Proof. Let $L = (a^n)^*$ and let A be a DFA recognizing L with n states. We give a fooling set of size $n + k$ for $E(L(A), d_p, k)$. We define $S \subseteq \Sigma^* \times \Sigma^*$ to consist of the following pairs of strings

1. $(a^n b^i, b^{k-i})$, for $1 \leq i \leq k$,
2. $(a^i, a^{n-i} b^k)$, for $1 \leq i \leq n$.

Let $i < j$ and consider two pairs $(X_i, Y_i) = (a^n b^i, b^{k-i})$ and $(X_j, Y_j) = (a^n b^j, b^{k-j})$. Then $X_j \cdot Y_i$ is not in $E(L, d_p, k)$ since $d_p(X_j \cdot Y_i, L) = |b^{k-i+j}| > k$. Now let $(X_i, Y_i) = (a^i, a^{n-i} b^k)$ and $(X_j, Y_j) = (a^j, a^{n-j} b^k)$. Then $X_j \cdot Y_i$ is not

in $E(L, d_p, k)$ since $d_p(X_j \cdot Y_i, L) = |a^{j-i}b^k| > k$. Finally, let $(X_i, Y_i) = (a^n b^i, b^{k-i})$ and $(X_j, Y_j) = (a^j, a^{n-j}b^k)$. Then $X_i \cdot Y_j$ is not in $E(L, d_p, k)$ since $d_p(X_i \cdot Y_j, L) = |b^i a^{n-j} b^k| > k$. □

Lemma 7 uses a binary alphabet. The same lower bound could be reached also by a finite language over a unary alphabet. However, if using a unary alphabet it seems that the fooling set method is not straightforwardly applicable and we would need to use an ad hoc proof for the lower bound.

Theorem 8. *For a regular language $L \subseteq \Sigma^*$ recognized by an NFA with n states and an integer $k \geq 0$,*

$$\mathrm{nsc}(E(L, d_p, k)) \leq n + k.$$

There exists a DFA A over a binary alphabet and with n states such that for all $k \geq 0$,

$$\mathrm{nsc}(E(L(A), d_p, k)) = n + k.$$

We get the results for the suffix distance neighbourhoods as a corollary of Theorem 8 and the observation that, for all strings x and y, $d_s(x, y) = d_p(x^R, y^R)$.

Corollary 9. *Let $k \geq 0$ and L be a regular language recognized by a DFA with n states. Then there is an NFA recognizing $E(L, d_s, k)$ with at most $n+k$ states.*

Proof. To see this, we take the original n-state NFA A for L and construct an NFA B for the language L^R simply by reversing the transitions and interchanging the sets of start states and accepting states. The NFA B has n states. We follow the construction from Proposition 6, which gives us an NFA A' with $n + k$ states recognizing the neighbourhood $E(L^R, d_p, k)$. Take the reverse of A', and we have an NFA A'' with $n + k$ states recognizing the neighbourhood $E(L, d_s, k)$. □

The following lemma is a symmetric variant of the lower bound construction for prefix distance neighbourhoods. As a consequence of Corollary 9 and Lemma 10 we then get a tight bound for the nondeterministic state complexity of suffix neighbourhoods.

Lemma 10. *For $n, k \in \mathbb{N}$, there exists a DFA A with n states over $\Sigma = \{a, b\}$ such that any NFA for $E(L(A), d_s, k)$ requires $n + k$ states.*

Proof. Let $L = (a^n)^*$ and let A be a DFA recognizing L with n states. We give a fooling set of size $n + k$ for $E(L(A), d_s, k)$. We define $S \subseteq \Sigma^* \times \Sigma^*$ to consist of the following pairs of strings

1. $(b^i, b^{k-i}a^n)$, for $1 \leq i \leq k$,
2. $(b^k a^i, a^{n-i})$, for $1 \leq i \leq n$.

Note that S is just the reverse of the fooling set from the proof of Lemma 7 and the result follows by symmetry. □

Theorem 11. *For a regular language $L \subseteq \Sigma^*$ recognized by an NFA with n states and an integer $k \geq 0$,*

$$\mathrm{nsc}(E(L, d_s, k)) \leq n + k.$$

There exists a DFA A defined over a binary alphabet with n states such that for all $k \geq 0$,

$$\mathrm{nsc}(E(L(A), d_s, k)) = n + k.$$

4.2. Substring Distance Neighbourhoods

A neighbourhood with respect to the substring distance can be recognized by an NFA that, roughly speaking, consists of $k+1$ copies of the NFA A recognizing the original language. Later we will show that the construction is optimal.

Lemma 12. *If A is an n-state NFA and $k \in \mathbb{N}_0$, the neighbourhood $E(L(A), d_f, k)$ can be recognized by an NFA with $(k+1) \cdot n + 2k$ states.*

Proof. Let $A = (Q, \Sigma, \delta, Q_0, F_A)$. We construct an ε-NFA $B = (P, \Sigma, \gamma, P_0, F_B)$ where

$$P = Q \times \{0, 1, \ldots, k\} \cup \{r_0, \ldots, r_{k-1}, s_1, \ldots, s_k\},$$

the set of initial states is $P_0 = \{r_0\}$, and the set of final states is

$$F_B = \{(q, i) \mid q \in F_A,\ 0 \leq i \leq k\} \cup \{s_1, \ldots, s_k\}.$$

There are three types of states.

- States r_i, $0 \leq i \leq k-1$, are reached in the first k steps of computation. The state r_i is reached by a prefix of length exactly i and has transitions to states in the i through kth copies of Q.
- States $(i, j) \in Q \times \{0, 1, \ldots, k\}$ are copies of the original state set Q. A word u reaches (i, j) if there exists a word v such that $i \in \delta(Q_0, v)$ and $d_s(u, v) = j$. States in $Q \times \{j\}$ are reached by states r_i when $i \leq j$ and "remember" the penalty j that may have been incurred at the beginning of the computation.
- States s_i, $1 \leq i \leq k$, are reached on a transition on any symbol from a state $(q, j) \in Q \times \{0, \ldots, k\}$ with $i = \varphi_A(q) + j + 1$. These states are similar to the states p_i from the construction of the NFA for the prefix neighbourhood presented in the proof of Proposition 6. In this case, we also have the addition of the penalty j that was incurred at the beginning of the computation.

For defining the transitions of B, recall from (1) that for a state $q \in Q$ of an NFA A, the value $\varphi_A(q)$ is the shortest length of a string w that reaches a final state of A from the state q. The transition function $\gamma : P \times (\Sigma \cup \{\varepsilon\}) \to 2^P$ is defined by setting

(i) for $b \in \Sigma$, $q \in Q$, $0 \leq j \leq k$,

$$\gamma((q,j),b) = \{(p,j) \mid p \in \delta(q,b)\};$$

(ii) for $0 \leq i \leq k-1$,

$$\gamma(r_i,\varepsilon) = \{(q,i+m) \mid (\exists w \in \Sigma^m, i+m \leq k, q_0 \in Q_0)\ q \in \delta(q_0,w)\};$$

(iii) for $b \in \Sigma$, $\gamma(r_i,b) = \{r_{i+1}\}$ for $0 \leq i \leq k-2$, and,

$$\gamma(r_{k-1},b) = \{(q,k) \mid q \in Q_0\};$$

(iv) for $q \in Q$, $0 \leq j \leq k$, $\gamma((q,j),\varepsilon) = s_{j+\varphi_A(q)}$ if $j + \varphi_A(q) \leq k$;
(v) for $1 \leq i \leq k-1$, $\gamma(s_i,b) = s_{i+1}$.

All transitions not listed above are undefined. Note that within a set of states $Q \times \{j\}$, $0 \leq j \leq k$, the transitions on symbols of Σ directly simulate the transitions of the original NFA A. Any symbol of Σ takes r_i, $0 \leq i \leq k-2$ (respectively, s_j, $1 \leq j \leq k-1$) to state r_{i+1} (respectively, s_{j+1}). Additionally, there is a transition on each $b \in \Sigma$ from r_{k-1} to states (q,k) where q is directly reachable from an initial state in the NFA A. Thus, a state r_i, $0 \leq i \leq k-1$, can be reached by any string of length exactly i and the state (q_0,k) is reached by all strings of length k. The ε-transitions take the states r_i, $0 \leq i \leq k-1$, to states in $Q \times \{0,1,\ldots k\}$ and the latter states to the states s_j, $1 \leq j \leq k$.

Consider $w \in E(L(A),d_f,k)$ where $w = u_1zu_2$ and $v_1zv_2 \in L(A)$, $u_i, v_i, z \in \Sigma^*$, $i = 1,2$, and $|u_1|+|v_1|+|u_2|+|v_2| \leq k$. Let C_0 be an accepting computation of A on v_1zv_2, beginning in state $q_0 \in Q_0$, and having state q_1 (respectively, q_2) after reading the prefix v_1 (respectively, the prefix v_1z).

Based on the computation C_0 we construct an accepting computation C_w of B on the input w. If $u_1 = v_1 = \varepsilon$, C_w begins in the state $(q_1,0)$ (and in this case $q_1 = q_0 \in Q_0$). If $u_1 = \varepsilon$ using an ε-transition (ii) the computation C_w takes the initial state r_0 to state $(q_1,|v_1|)$. If $u_1 \neq \varepsilon$ and $|u_1| < k$, beginning in state r_0 and C_w reads the prefix u_1 and end in state $r_{|u_1|}$. Then using an ε-transition (ii) C_w reaches the state state $(q_1,|u_1|+|v_1|)$.

Beginning in state $(q_1,|u_1|+|v_1|)$ computation C_w reads the substring z directly simulating in the first component the computation C_0. Thus, after prefix u_1z the computation C_w is in state $(q_2,|u_1|+|v_1|)$. Since A accepts the suffix v_2 when the computation begins in state q_2, it follows that $\varphi_A(q_2) \leq |v_2|$. Thus according to rules (iv), the state $(q_2,|u_1|+|v_1|)$ has an ε-transition to state $s_{|u_1|+|v_1|+h}$ where $h \leq |v_2|$. Since $|u_1|+|v_1|+|u_2|+|v_2| \leq k$, after reading the suffix u_2, the computation C_w ends in the accepting state $s_{|u_1|+|v_1|+h+|u_2|}$.

Conversely to verify that $L(B) \subseteq E(L(A),d_f,k)$ assume that B has an accepting computation C_w on string w. Let (q,j) be the first state in $Q \times \{0,1,\ldots,k\}$ that is reached in C_w after reading a prefix u of w. Since a state r_i, $0 \leq i \leq k-1$, is reached exactly by the strings of length i, it follows from the definition of the ε-transitions (ii) and the transition on Σ from state r_k that the state q can be reached from an initial state of A with a string v where $|u|+|v| = j$.

Let z be the substring after u that is processed using transitions (i). Thus, the computation reaches the end of the prefix uz in a state (q', j), and let u' be the suffix of the input (that is, $w = uzu'$). By the definition of the ε-transitions and the choice of the final states of B it follows that $j + \varphi_A(q') + u' \leq k$ and, recalling $j = |u| + |v|$ it follows that $w \in E(L(A), d_f, k)$.

The ε-NFA B has $(k+1) \cdot n + 2k$ states. It follows that there exists an equivalent ordinary NFA with the same number of states. □

Lemma 13. *There exists a DFA A with n states such that, for all $k \geq 0$,*

$$\mathrm{nsc}(E(L(A), d_f, k)) \geq (k+1) \cdot n + 2k.$$

Proof. Choose $\Sigma = \{a, b\}$ and $L = (a^n)^*$. The minimal incomplete DFA for L has n states.

Next we construct a fooling set of cardinality $(k+1) \cdot n + 2k$ for $E(L, d_f, k)$ assuming that $k \leq n$ and at the end of the proof we explain how the fooling set can be modified to handle the case $n < k$.

Define

$$S_1 = \{(b^\ell a^i,\ a^{n-i} b^{k-\ell}) \mid 0 \leq i \leq n-1,\ 0 \leq \ell \leq k\},$$

$$S_2 = \{(a^n b^j,\ b^{k-j}) \mid 1 \leq j \leq k\},\ \ S_3 = \{(b^j,\ b^{k-j} a^n) \mid 1 \leq j \leq k\}.$$

We note that $|S_1 \cup S_2 \cup S_3| = (k+1) \cdot n + 2k$ and, by Proposition 1, it is sufficient to show that the set $S = S_1 \cup S_2 \cup S_3$ is a fooling set for L.

For every $(x, y) \in S$, $x \cdot y$ is of the form $b^i a^n b^j$ where $i + j = k$. This means that $d_f(x \cdot y, L) = d_f(x \cdot y, a^n) = k$. It remains to verify the second condition in the definition of fooling sets holds for all two distinct elements of S.

First consider two distinct pairs in S_1, $(b^\ell a^i, a^{n-i} b^{k-\ell})$ and $(b^{\ell'} a^{i'}, a^{n-i'} b^{k-\ell'})$ where $i \neq i'$ or $\ell \neq \ell'$. First assume that $\ell \neq \ell'$ and, without loss of generality $\ell < \ell'$. Now the string $b^{\ell'} a^{i'} \cdot a^{n-i} b^{k-\ell}$ has subword distance at least $\ell' + k - \ell > k$ to any string in L. The remaining possibility is then that $\ell = \ell'$ and $i < i'$. Now the string $b^{\ell'} a^{i'} \cdot a^{n-i} b^{k-\ell}$ has subword distance $\min(i' - i,\ n - i' + i) + k$ to the closest string in L (which is either a^n or a^{2n}).

Next consider two distinct pairs in S_2, $(a^n b^j, b^{k-j})$ and $(a^n b^{j'}, b^{k-j'})$, where $j < j'$. The concatenation $a^n b^{j'} \cdot b^{k-j}$ has subword distance $k + j' - j > k$ to L. The case where we have two distinct pairs in S_3 is completely analogous.

Next consider $(b^\ell a^i,\ a^{n-i} b^{k-\ell}) \in S_1$ and $(a^n b^j, b^{k-j}) \in S_2$. The string $a^n b^j \cdot a^{n-i} b^{k-\ell}$ can be in $E(L, d_f, k)$ only if the length of $b^j \cdot a^{n-i} b^{k-\ell}$ is at most k, which, in particular, implies that $j < \ell$ since $n - i$ cannot be zero (as $i \leq n-1$). Now if $j < \ell$, the string $b^\ell a^i \cdot b^{k-j}$ has distance at least $\ell + k - j > k$ from a string in L.

The case where one of the pairs is in S_1 and the other is in S_3 is completely analogous to the previous one.

As the last possibility consider $(a^n b^j, b^{k-j}) \in S_2$ and $(b^i, b^{k-i} a^n) \in S_3$, and consider the concatenation $w = a^n b^j \cdot b^{k-i} a^n$. Since $j \geq 1$ the longest subwords w shares with a word of L is either the prefix a^n or the suffix a^n and the subword

distance to L is $n + j + k - i$, and under our original assumption $k \leq n$, this means that the subword distance to L is greater than k.

In the proof we have used the assumption $k \leq n$ only in the last case. In the case $n < k$, we can modify the definition of S_2 and S_3 by replacing there the occurrences of a subword a^n with $a^{m \cdot n}$ where $m \cdot n > k$. After this modification, in the last case, the string $a^{m \cdot n} b^j \cdot b^{k-i} a^{m \cdot n}$ is guaranteed to have subword distance greater than k to L also in case $n < k$. □

As a consequence of Lemmas 12 and 13 we have an exact bound for the nondeterministic state complexity of neighbourhoods with respect to the substring distance:

Theorem 14. *If L has an NFA with n states and $k \in \mathbb{N}_0$,*

$$\mathrm{nsc}(E(L, d_f, k)) \leq (k+1) \cdot n + 2k.$$

For every $n \in \mathbb{N}$ there exists a DFA A over a binary alphabet and with n states such that for all $k \in \mathbb{N}_0$, $\mathrm{nsc}(E(L(A), d_f, k)) = (k+1) \cdot n + 2k$.

5. Conclusion

We have given a tight bound for the deterministic state complexity of neighbourhoods with respect to the prefix distance and tight bounds for the nondeterministic state complexity of the prefix, suffix and substring distance neighbourhoods.

Due to the fact that the reversal of a regular language L can be recognized by an NFA having the same size as an NFA for L, the bounds for the nondeterministic state complexity of suffix neighbourhoods were obtained as a corollary of the corresponding bounds for prefix neighbourhoods. The situation is essentially different for DFAs since, for a DFA A with n states, the incomplete DFA recognizing $L(A)^R$ needs in the worst case $2^n - 1$ states. It seems likely that constructing a DFA for the neighbourhood of an n-state DFA with respect to the suffix distance causes a much larger worst-case size blow-up than the bound for prefix distance in Proposition 2. Obtaining tight bounds for the deterministic state complexity of neighbourhoods with respect to the suffix distance, or the substring distance, remains an open problem.

[1] Apostolico, A.: Maximal Words in Sequence Comparisons Based on Subword Composition. In: Elomaa, T. et al. (Eds.), Ukkonen Festschrift, Lect. Notes Comput. Sci. 6060 (2010) 34–44

[2] Birget, J.C.: Intersection and union of regular languages and state complexity. Information Processing Letters **43** (1992) 185–190

[3] Calude, C.S., Salomaa, K., Yu, S.: Additive Distances and Quasi-Distances Between Words. Journal of Universal Computer Science **8**(2) (2002) 141–152

[4] Choffrut, C., Pighizzini, G.: Distances between languages and reflexivity of relations. Theoretical Computer Science **286**(1) (2002) 117–138

[5] Deza, M.M., Deza, E.: Encyclopedia of Distances. Springer Berlin Heidelberg (2009)

[6] Gao, Y., Moreira, N., Reis, R., Yu, S.: A survey on operational state complexity. `arXiv:1509.03254v1 [cs.FL]`, Sept. 2015. To appear in *Computer Science Review.*

[7] Holzer, M. Kutrib, M.: Descriptional and computational complexity of finite automata — A survey. *Inform. Comput.* **209** (2011) 456–470.

[8] Kari, L., Konstantinidis, S.: Descriptional complexity of error/edit systems. Journal of Automata, Languages, and Combinatorics **9** (2004) 293–309

[9] Kari, L., Konstantinidis, S., Kopecki, S., Yang, M.: An efficient algorithm for computing the edit distance of a regular language via input-altering transducers. `CoRR abs/1406.1041` (2014)

[10] Konstantinidis, S.: Computing the edit distance of a regular language. Information and Computation **205** (2007) 1307–1316

[11] Kutrib, M., Meckel, K., Wendlandt, M.: Parameterized Prefix Distance between Regular Languages. In: SOFSEM 2014: Theory and Practice of Computer Science. Lect. Notes Comput. Sci. **8327** (2014), Springer, 419–430

[12] Kutrib, M., Pighizzini, G.: Recent trends in descriptional complexity of formal languages. *Bulletin of the EATCS* 111 (2013) 70–86.

[13] Lothaire, M.: Applied Combinatorics on Words, Ch. 1 Algorithms on Words. Encyclopedia of Mathematics and It's Applications 105, Cambridge University Press, New York, 2005

[14] Ng, T., Rappaport, D., Salomaa, K.: Quasi-Distances and Weighted Finite Automata. In: Descriptional Complexity of Formal Systems, DCFS'15, Waterloo, Ontario, June 25–27, 2015, Lect. Notes Comput. Sci. **9118** (2015) 209–219

[15] Ng, T., Rappaport, D., Salomaa, K.: State complexity of neighbourhoods and approximate pattern matching. In: Developments in Language Theory, DLT 2015, Liverpool, UK, July 27–30, Lect. Notes Comput. Sci. 9168 (2015) 389–400

[16] Povarov, G.: Descriptive Complexity of the Hamming Neighborhood of a Regular Language. In: Language and Automata Theory and Applications. (2007) 509–520

[17] Salomaa, K., Schofield, P.: State Complexity of Additive Weighted Finite Automata. International Journal of Foundations of Computer Science **18**(06) (December 2007) 1407–1416

[18] Shallit, J.: A second course in formal languages and automata theory. Cambridge University Press, Cambridge, MA (2009)

[19] Yu, S.: Regular languages. In Rozenberg, G., Salomaa, A., eds.: Handbook of Formal Languages. Springer-Verlag, Berlin, Heidelberg (1997) 41–110