Cybernetics and Systems Analysis, Vol. 43, No. 4, 2007

VECTOR REPRESENTATIONS FOR EFFICIENT COMPARISON AND SEARCH FOR SIMILAR STRINGS

A. M. Sokolov

UDC 004.032.26+004.422.632.2+004.424.4

A method is proposed for approximation of the classic edit distance between strings. The method is based on a mapping of strings into vectors belonging to a space with an easily calculable metric. The method preserves the closeness of strings and makes it possible to accelerate the computation of edit distances. The developed q-gram method of approximation of edit distances and its two randomized versions improves the approximation quality in comparison with well-known results.

Keywords: *approximating edit distance, metric embedding, approximate nearest-neighbor search, neural-network-based information technologies.*

INTRODUCTION

The classical edit distance ed (x, y) between symbolic strings x and y (the Levenstein distance [1] or the minimal number of operations of insertion, replacement, and elimination of symbols required for transformation of x into y) is used in many fields beginning with genetics and web search and ending with speech recognition. Since present-day amounts of string data are enormous and their exponential growth continues [2], the classical $O(n^2)$-algorithm [3, 4] for computation of this distance between strings of length n is often practically inapplicable. This circumstance stipulated the emergence of new fields of research connected with the acceleration of computation or approximation of edit distance [5] and the use of methods of the theory of metric embeddings [6] for transformation of objects in spaces in which the computation of complicated initial metrics is simplified.

A topical problem of edit distance approximation is the search for the embedding of the Levenstein metric into a vector space [7]. The existing methods of approximating the Levenstein distance and searching for similar strings in large arrays should be improved with a view to increasing their computational efficiency, decreasing requirements to resources, and increasing the approximation accuracy.

In the present article, a method of embedding strings into a vector space is developed that is based on the use of distributed neural-network representations [8, 9]. We propose a deterministic method (Sec. 3) for embedding edit distance into a Manhattan space and also randomized versions of this nearest-neighbor search method (Sec. 4).

1. DEFINITIONS AND AUXILIARY STATEMENTS

We denote by Σ an alphabet of symbols and by Σ^n a set of strings of length $n \in \mathbb{N}$ that are specified over Σ. We denote a symbol located at the ith position of a string x by $x[i]$. We denote a substring $x[i]x[i+1]\ldots x[j]$ of a string x by $x[i, j]$, write $x[i, j] \subseteq x$, and call ranges of positions of the form $[i, j]$ intervals. We call substrings of the form $x[i, i+q-1]$, $q \in \mathbb{N}$, q-grams and the set of all q-grams of a string its q-spectrum. The length of a string x is denoted by $|x|$. For $x \in \Sigma^n$ and $q \in \mathbb{N}$,

---

International Scientific-Educational Center of Information Technologies and Systems, Kiev, Ukraine, *sokolov@ukr.net*. Translated from Kibernetika i Sistemnyi Analiz, No. 4, pp. 18–38, July–August 2007. Original article submitted November 30, 2005.

we call a q-gram vector of the string a vector $v_{n,q}(x) \in (\mathbb{N} \cup \{0\})^{|\Sigma|}$ in which to each q-gram $\sigma \in \Sigma^q$ corresponds an element of a vector $(v_{n,q}(x))_\sigma \in \mathbb{N} \cup \{0\}$ and this element is equal to the number of occurrences of σ in x: $(v_{n,q}(x))_\sigma = \sum_{i=1}^{n-q+1} [x[i, i+q-1] = \sigma]$. We write $v_q(x)$ or simply $v(x)$ instead of $v_{n,q}(x)$ when this will not lead to ambiguity. For a pair of strings $x, y \in \Sigma^n$, the q-gram distance $d_q(x, y)$ is understood to be the Manhattan distance (the l_1-distance) between the corresponding q-gram vectors $||v_q(x) - v_q(y)||_{l_1} = \sum_{\sigma \in \Sigma^q} |(v_q(x))_\sigma - (v_q(y))_\sigma|$. A sequence of edit operations that transform a string x into y is called a reconstruction of y from x.

By a de Bruijn graph [10, 11] $B(\Sigma; q)$ for the alphabet Σ and a parameter $q \geq 3$ we understand a directed graph $G(V, E)$ to the set of nodes V of which correspond all $(q-1)$-grams and to the set of edges $E \subset V \times V$ correspond all q-grams in this alphabet. We call q- and $(q-1)$-grams corresponding to edges and nodes labels. In this case, for symbols $l_i \in \Sigma$, the edge with a label $l_1 l_2 \ldots l_q$ connects the nodes with labels $l_1 l_2 \ldots l_{q-1}$ and $l_2 l_3 \ldots l_q$. In a de Bruijn graph, to each string x, $|x| \geq q$, corresponds a definite path π_x consisting of the edges that successively connect the nodes whose labels are $(q-1)$-grams that successively enter in the string. We call a subpath π'_x of the path π_x a connected sequence of edges that corresponds to a substring $x' \subseteq x$ in this path and denote $\pi'_x \subseteq \pi_x$. We denote a path π_x consisting of two subpaths π'_x and π''_x by $\pi'_x \pi''_x$.

For x, $|x| \geq q$, we denote by $B[x; q]$ the subgraph of a de Bruijn graph whose nodes are all the $(q-1)$-grams of the string x (the elements of the $(q-1)$-spectrum of x) and edges correspond to its q-grams. We denote by $B[x, y; q]$ the graph constructed over the union of $(q-1)$-spectra of two strings $x, y \in \Sigma^w$ of identical length w.

A string x, $|x| \geq w$ is called (q, w)-nonrepetitive if, in any interval consisting of w symbols, all the $w - q + 1$ q-grams are different. A string $x \in \Sigma^w$ that is (q, w)-nonrepetitive is called simply q-nonrepetitive.

Let us consider two cases when repetitions of q-grams take place.

Case 1. Both strings $x, y \in \Sigma^w$ are q-nonrepetitive.

Case 2. At least one of strings $x, y \in \Sigma^w$ is not q-nonrepetitive.

Let us consider case 1. If x and y contain a common q-gram with labels $l_1 \ldots l_q$, then the paths π_x and π_y corresponding to them in $B[x, y; q]$ pass through the same edge connecting the nodes labelled by $l_1 \ldots l_{q-1}$ and $l_2 \ldots l_q$. By left (of the form $-\!<$) and right (of the form $>\!-$) branchpoints we understand nodes at which the paths π_x and π_y, respectively, diverge after a common edge or converge before a common edge. In this case, adjacent fragments of the form $>\!<$, $-\!<$, and $>\!-$, i.e., those containing at least one common edge of both paths π_x and π_y and bounded from one or both sides by a branchpoint, can be selected in $B[x, y; q], x \neq y$.

For case 1, we introduce the concepts of a loop (a half-loop), a fork (a half-fork), and a shift. For each of two paths, the collection consisting of the left branchpoint and that nearest to it in the order of the traverse of edges by the right branchpoint and also edges of this path that are between them (of the form $<\!->$) is called its half-loop. To each half-loop corresponds a half-loop (its length can be zero) consisting of edges of the second path and connecting the same branchpoints (then these half-loops taken together are of the form $<\!>$) or, in the case of a configuration of the form $>\!<$ (if the left and right points of two such adjacent configurations are connected by a half-loop of the form $>\!<\!->\!<$) corresponds the half-loop connecting the remained last left branchpoint and the first right branchpoint that is next to it. We call such a pair of half-loops corresponding to one another a loop. According to this definition, at least one of half-loops of a loop has a nonzero length.

We call a graph $B[x, y; q]$ a fork if there exists a subpath π_c, $w > |\pi_c| > 0$, that is common to π_x and π_y and is such that we have $\pi_x = \pi'_x \pi_c \pi''_x$ and $\pi_y = \pi'_y \pi_c \pi''_y$ and in which, for any edges $e \in \pi'_x \cup \pi''_x$ and $e' \in \pi'_y \cup \pi''_y$, the condition $e \neq e'$ is satisfied. Since we have $|x| = |y|$, we obtain $|\pi'_x| + |\pi''_x| = |\pi'_y| + |\pi''_y|$. The subpaths π'_x, π''_x, π'_y, and π''_y of the left or right forks composed of edges of one string are called half-forks. By a shift we understand a graph that is a special case of a fork graph in which there exists a subpath π_c, $w \geq |\pi_c| > 0$, that is common to π_x and π_y and is such that we have $\pi_x = \pi'_x \pi_c$, $\pi_y = \pi_c \pi'_y$, and, $\forall\, e \in \pi'_x$, $e' \in \pi'_y$, we have $e \neq e'$. In this case, we also call subpaths π'_x and π'_y half-forks. If $B[x, y; q]$ is a shift, then we say that x is a shift of y and, vice versa, y is a shift of x.

The statements formulated below are true.

Statement 1. If $B[x, y; q]$ is a fork (a shift), then $B[x, y; q+1]$ also is a fork (a shift) with the same number of edges in half-forks.

Statement 2. If $B[x, y; q]$, $x, y \in \Sigma^n$, is a fork with the length of the common fragment equal to $|\pi_c|$ and, at the same time, the right fork consists of two half-forks of length $|\pi'_x| = s_1$ and $|\pi'_y| = s_2$ and the left one consists of two half-forks of length $|\pi''_x| = s_3$ and $|\pi''_y| = s_4$, then the transformation of x into y requires no more than $\min(s_1, s_2) + \min(s_3, s_4)$ replacement operations and $|s_2 - s_1| + |s_4 - s_3|$ operations of insertion or elimination. Since we have $s_3 = n - |\pi_c| - s_1$, $s_4 = n - |\pi_c| - s_2$, in the aggregate, we obtain $\mathrm{ed}(x, y) \le \max(s_1, s_2) + \max(s_3, s_4)$. Note that the upper estimate is also true for x and y whose graph is not a fork but π_x and π_y have some common fragment π_c and π'_x and π'_y (or π''_x and π''_y) can be partially coincident.

Case 2. When strings are not (q, w)-nonrepetitive in the graph $B[x; q]$ ($B[y; q]$), one or both substrings have at least one subpath $\pi'_x \subseteq \pi_x$ ($\pi'_y \subseteq \pi_y$) corresponding to a substring $x' \subseteq x$ ($y' \subseteq y$) that has self-intersection and, hence, that is a cycle. This cycle has at least one node occurring more than once and, consequently, at least one repetitive $(q-1)$-gram. It may be noted that the same cycle can correspond to different substrings x' (y'). To this end, it suffices to begin the traverse of such a cycle with another node.

The lemma presented below asserts that, for a sufficiently large q, the graph $B[x; q]$, where x is a (q, w)-nonrepetitive string, contains no more than one cycle.

LEMMA 1. For $x \in \Sigma^w$ and $q > 2w/3$, if there is a substring $x' \subseteq x$ such that $\pi_{x'}$ forms a cycle C in $B[x; q]$, then identical edges outside of the cycle C are absent.

Proof. Let $f = |x'|$, and let substrings $x'' = x[i, i+q-2]$ and $x''' = x[j, j+q-2]$, $i < j$, be a pair of $(q-1)$-grams that are maximally distant among the coincident ones in $\pi_{x'}$. Since we have $q > 2w/3$, at least $2[2w/3] - w$ symbols of these $(q-1)$-grams intersect, and the symbols of a substring $x[i, j-1]$ will be periodically repeated successively in x'. If we denote the symbols of the substring $x[i, j-1]$ by $b_1 b_2 \ldots b_B$, then we have $x'[k] = b_{((k-1) \bmod B)+1}$, $k = 1, \ldots, f$.

We assume that two $(q-1)$-grams $x[i', i'+q-2] = x[j', j'+q-2]$, $j' > i' > j$, i.e., $(q-1)$-grams whose right ends extend beyond the right boundary of x''', coincide. Since we have $q > 2w/3$, these q-grams completely cover the fragment that is common to x'', x''' and contains at least one occurrence of the entire sequence of symbols $b_1 \ldots b_B$. Since a similar reasoning concerning the periodic character (with its repetitive sequence) is valid for the substring $x[i', j'+q-2]$, $\pi_{x[i, j+q-2]}$ and $\pi_{x[i', j'+q-2]}$ pass through the same edges as the cycle C. The case of coincidence of $(q-1)$-grams before x'' is similarly considered.

LEMMA 2. For $x, y \in \Sigma^w$ and $q > 2w/3$, if the first (last) nodes of subpaths $\pi_{x'} \subseteq \pi_x$ ($\pi_{y'} \subseteq \pi_y$) forming a common cycle C do not coincide, then the common edges of π_x and π_y are absent before (after) these nodes.

Proof. Let us consider the case when the subpaths $\pi_{x'}$ and $\pi_{y'}$ consisting of edges of the cycle C terminate at different nodes of the cycle. Let $x[i, i+q-1]$ and $y[j, j+q-1]$ be the q-grams corresponding to the last edges of the mentioned subpaths in the cycle C. Assume that, for $i' > i$, $j' > j$, two q-grams coincide, $\tilde{x} = x[i', i'+q-1] = y[j', j'+q-1] = \tilde{y}$. Since we have $q > 2w/3$, they contain at least one sequence of symbols $b_1 \ldots b_B$ mentioned in the proof of Lemma 1.

Let the positions of the right ends of $x[i, i+1-1]$ and $y[j, j+q-1]$ within $\tilde{x}$ and $\tilde{y}$ be i'' and j'', respectively. Since subpaths terminate at different nodes and we have $\tilde{x} = \tilde{y}$, we obtain $i'' \ne j''$.

Assume that we have $i'' < j''$. It follows from the truth of the inequality $\tilde{x} = \tilde{y}$ that we have $\tilde{x}[i''+1, j''] = \tilde{y}[i''+1, j'']$, which is equivalent to the truth of the equality $x[i+q, i+q-1+(j''-i'')] = y[j+q-(j''-i''), j+q-1]$. Since y' and x' are composed of periodically repetitive sequences of symbols $b_1 \ldots b_B$, first, the last symbol of x' is not the q-gram $x[i, i+q-1]$ but $x[i+(j''-i''), i+q-1+(j''-i'')]$ and, second, x' and y' terminate at the same node, contrary to the condition of the lemma.

The following statement describes the behavior pattern of paths in graphs with cycles when q varies.

Statement 3. If, for $x \in \Sigma^w$, there is a cycle with repetitive edges in $B[x; q]$, $q > 2w/3$, then the length of the path fragment between these nodes that passes more than once along the same edges is reduced in all cases when q is incremented by unity. In this case, the total number of unique edges in the cycle is not changed. If repetitive edges are absent in the cycle (i.e., each edge of the cycle is present in π_x exactly once), then the cycle vanishes after the next increment of q by unity.

Proof. By Lemma 1, the graph $B[x; q]$ contains a unique cycle. This statement can be verified by tracing the change in the number of edges within the cycle and outside of it as a result of the successive increment in q in the graph $B[x; q]$ with one cycle.

2. LOWER AND UPPER CONSTRAINTS ON EDIT DISTANCE

It is necessary to find a definition of the distance between strings x and y and a threshold of its value that are such that, for distances that are shorter than this threshold (under some conditions that are imposed on definite values and do not depend on strings), in case 1, the graph $B[x, y; q]$ would contain a shift or a fork and would not contain half-loops. This would make it possible to reconstruct strings, using Statement 2 and a small number of operations (the less this number, the exacter approximation of edit distance). There are distances that do not satisfy these requirements. In particular, if the usual q-gram distance (the l_1-distance between q-gram vectors) and any admissible fixed q are used, then there are strings x and y such that the graph $B[x, y; q]$ contains a loop. In this case, the sought-for threshold cannot be found.

In case 1 when strings are (q, w)-nonrepetitive, with increasing q for half-loops and forks, there are differences in dependences between the numbers of different q-grams. Therefore, to determine the difference between a half-loop and a fork, we propose to use the following distance based on the usual q-gram distance:

$$d^{\Sigma}_{w, q_1, q_2}(x, y) = \sum_{q=q_1}^{q_2} d_q(x, y) \tag{1}$$

defined for strings of identical length w and fixed values of q_1 and q_2 for the lengths of q-grams. Let us show that, with the help of this distance, the presence of a fork or a shift can be found, i.e., the possibility of reconstruction of a pair of strings with the help of a small number of edit operations. Case 2 in which repetitions of q-grams take place should be individually considered (Lemma 4).

We denote $\Delta q = q_2 - q_1$ and $Q = (\Delta q + 1)(\Delta q + 2)$. We call a pair of strings x and y bad if the inequality $d^{\Sigma}_{w, q_1, q_2}(x, y) < Q$ is not fulfilled and good otherwise. Let us show that, for a good pair of strings x and y, the graph $B[x, y; q_2]$ is a fork (or a shift).

In Lemma 3, the case is considered when, for definite values of q_1, q_2 $(q_2 > q_1)$, and w, strings x and y are (q_1, w)-nonrepetitive, and necessary conditions of existence of a loop in the graph $B[x, y; q_2]$ are found (in this case, the pair x, y is bad). Their nonfulfillment is the sufficient condition of the presence of a shift or a fork if common cycles are absent.

LEMMA 3. Let $x, y \in \Sigma^w$ be (q_1, w)-nonrepetitive, and let the following inequalities be fulfilled: $w \geq q_2 > q_1 \geq 3$, $\Delta q \leq \left\lfloor \dfrac{w - q_1 + 1}{2} \right\rfloor$, and

$$Q \leq 4(w - q_2 + 1), \tag{2}$$

$$d^{\Sigma}_{w, q_1, q_2}(x, y) < Q. \tag{3}$$

Then we have $\mathrm{ed}(x, y) < 2(\Delta q + 1)$.

Proof. For any half-loop that has at least one common edge before and after the left and right branchpoints, respectively, in the graph $B[x, y; q]$, the increment of q by unity leads to the increment of the number of edges by unity in each half-loop that forms a loop from each string. Since, by the definition of half-loops in the graph $B[x, y; q]$, edges of one half-loop of one string do not coincide with edges of the half-loop corresponding to it in the other string, coincidences of loop edges in $B[x, y; q + 1]$ are all the more absent since the labels of edges in the graph $B[x, y; q + 1]$ contain labels of edges of the graph $B[x, y; q]$ as substrings, $x[i, i + q - 1] \to x[i, i + q] = x[i, i + q - 1]\, x[i + q]$. Therefore, for strings x and y containing at least one loop surrounded by common edges, the inequality $d_{q+1}(x, y) \geq d_q(x, y) + 2$ is fulfilled. But if $B[x, y; q]$ is a fork, then, by virtue of Statement 1, the fork is preserved after incrementing q by unity and $d_q(x, y)$ is not changed. In this case, the total number of edges decreases by unity.

Since any half-loop is not preserved when the value of Δq is larger than half the edges available in the graph $B[x, y; q_1]$, we impose the additional condition $\Delta q \leq \lfloor (w - q_1 + 1)/2 \rfloor$ to preserve half-loops.

Taking into account that the minimal q-gram distance between different strings equals two and on the condition that the right and left branchpoints of a loop do not become, respectively, the first or last node of paths in $B[x, y; q]$ when $q < q_2$ (which is provided by condition (2)), we obtain

$$d^{\Sigma}_{w, q_1, q_2}(x, y) \geq \sum_{q=q_1}^{q_2} (d_{q_1}(x, y) + 2(q - q_1)) \geq 2(q_2 - q_1 + 1) + (q_2 - q_1) \cdot (q_2 - q_1 + 1) = (\Delta q + 1)(\Delta q + 2) = Q. \tag{4}$$

To assert that, when the inequality $d^{\Sigma}_{w, q_1, q_2}(x, y) < Q$ is true, a configuration of the form = (coincident nodes in the paths π_y and π_x are absent) cannot arise, instead of configurations of a shift or a fork, the condition $|\pi_c| \geq 1$ included in the definition of a fork and a shift must be true, i.e., at least one common edge must be present for $q = q_2 - 1$: $d_{q_2-1}(x, y) \leq 2(w - (q_2 - 1) + 1) - 2 = 2(w - q_2 + 1)$. But if we have $d_{q_2-1}(x, y) > 2(w - q_2 + 1)$ (i.e., a common edge is absent in the graph $B[x, y; q_2 - 1]$), then we obtain $d^{\Sigma}_{w, q_1, q_2}(x, y) = \sum_{q=q_1}^{q_2-1} d_q(x, y) + d_{w,q_2}(x, y) > 4(w - q_2 + 1)$. Hence, if we have $d^{\Sigma}_{w,q_1,q_2}(x, y) \leq 4(w - q_2 + 1)$, then we obtain $d_{q_2-1}(x, y) \leq 2(w - q_2 + 1)$, and there is at least one common node when $q = q_2$.

Thus, it follows from the presence of loops in the graph $B[x, y; q]$ for $q = q_1, \ldots, q_2$ that we have $d^{\Sigma}_{w, q_1, q_2}(x, y) \geq Q$. But since condition (3) implies the truth of the inequality $d^{\Sigma}_{w, q_1, q_2}(x, y) < Q$, loops cannot be present in $B[x, y; q_2]$, i.e., $B[x, y; q_2]$ is a fork (a shift) or configurations of paths in $B[x, y; q_2]$ are of the form =. The latter situation is excluded by virtue of condition (2). Then the graph $B[x, y; q_2]$ is a fork (a shift).

According to Statement 2, to reconstruct a fork when q is fixed, it is necessary to perform no more than $\max(s_1, s_2) + \max(s_3, s_4) \leq s_1 + s_2 + s_3 + s_4 = d_q(x, y)$ edit operations. Since the value of $q = q_1, \ldots, q_2$ at which a fork is formed is not known in advance, one can only assert that we have ed $(x, y) \leq d_{q_2}(x, y)$. Next, for any q_2, a loop cannot be present since we have $d^{\Sigma}_{w, q_1, q_2}(x, y) < Q$. We denote by q^*, $q_1 \leq q^* < q_2$, the last value of q for which a loop is still available. Then we have $d^{\Sigma}_{w, q_1, q^*}(x, y) \geq (q^* - q_1 + 1)(q^* - q_1 + 2)$ and $Q > d^{\Sigma}_{w, q_1, q_2}(x, y) = d^{\Sigma}_{w, q_1, q^*}(x, y) + (q_2 - q^*) \times d_{q^*}(x, y) \geq (q^* - q_1 + 1)(q^* - q_1 + 2) + (q_2 - q^*) d_{q_2}(x, y)$, i.e., we obtain

$$\text{ed}(x, y) \leq d_{q_2}(x, y) < \frac{(Q - (q^* - q_1 + 1)(q^* - q_1 + 2))}{q_2 - q^*} = q_2 + q^* - 2q_1 + 3 \leq 2(\Delta q + 1).$$

We now consider case 2 in which repetitions of q-grams are present. Our interest is in the dependence of the number of common edges in $B[x, y; q]$ in two paths π_x and π_y on q. We note that only the situations of presence of cycles in which edges of one path coincide with edges of another path (in this case, a coincident edge occurs more than once in at least one cycle) and in which a decrease in the number of edges in Statement 3 does not compensate the increase in the distance $d_q(x, y)$ are different in this case from case 1. Therefore, we will restrict ourselves to the consideration of the cases when a cycle contained in a path π_x and a cycle contained in a path π_y are identical when $q = q_1$, but the lengths of subpaths $\pi_{x'} \subseteq \pi_x$ and $\pi_{y'} \subseteq \pi_y$ that belong to the cycle and its entrance and exit nodes can be different.

LEMMA 4. Let, when $q_1 > 2w/3$, for strings $x \subseteq y \in \Sigma^w$, there be substrings $x' \subseteq x$ and $y' \subseteq y$, $x' \neq y'$, such that the paths $\pi_{x'}$ and $\pi_{y'}$ that correspond to them in the graph $B[x, y; q]$ consist of edges belonging to their common cycle C. Let we also have $d^{\Sigma}_{w, q_1, q_2}(x, y) \leq Q$. Then we obtain ed $(x, y) < 2(\Delta q + 1)$.

Proof. Assume, by contradiction, that the inequality ed $(x, y) \geq 2(\Delta q + 1)$ is true. Let us show that, in this case, we have $d^{\Sigma}_{w, q_1, q_2}(x, y) > Q$. We find a minimally possible value of $d^{\Sigma}_{w, q_1, q_2}(x, y)$ under the conditions of the lemma. Since the character of changing $d_q(x, y)$ in the presence of cycles can be very complicated and various and the obtaining of this dependence in the closed form is cumbersome, we determine the set of parameters for which the sought-for minimum of $d^{\Sigma}_{w, q_1, q_2}(x, y)$ can be attained using the following qualitative considerations based on the comparison of dependences of the values of $d_q(x, y)$ on the character of the traverse of the cycle along both paths π_x and π_y.

Let S be the smaller number of edges of the cycle C between the nodes corresponding to the initial nodes (to the first $(q_1 - 1)$-grams) of paths $\pi_{x'} \subseteq \pi_x$ and $\pi_{y'} \subseteq \pi_y$. Let L_x and L_y be the lengths of the subpaths $\pi_{x'}$ and $\pi_{y'}$, respectively, that belong to the cycle C.

When one cycle is present, the behavior of $d_q(x', y')$ is completely determined by the mentioned parameters s, L, L_x, and L_y until we have $L_x > L$ and $L_y > L$ and, hence, $d^{\Sigma}_{w, q_1, q_2}(x', y')$ can be considered as a function of s, L, L_x, and L_y. With increasing the length a q-gram, the distance $d_q(x', y')$ changes along the lines of constant values of the

quantity $|L_x - L_y|$ with decrementing L_x and L_y by unity. If we have $L_x < L$ and $L_y < L$, then cycles in both graphs $B[x; q]$ and $B[y; q]$ vanish and, with increasing q, the further behavior of $d_q(x', y')$ is described by Lemma 3.

By Lemma 2 , edges of different paths can coincide outside of a cycle if the first or last nodes of the paths $\pi_{x'}$ and $\pi_{y'}$ coincide. The contribution of the edges of the paths outside of the cycle to $d_q(x, y)$ is no less than $|L_x - L_y|$ when $L_x \geq L$ and $L_y \geq L$ (which is obtained when $s = 0$ or when the last nodes of $\pi_{x'}$ and $\pi_{y'}$ coincide).Therefore, we have $d_q(x, y) \geq d_q(x', y') + |L_x - L_y|$.

We can show that, for $s > 0$, the sum of successive values of $d_q(x', y')$ along such a line is no less than for $s = 0$ for the same value of $|L_x - L_y|$ in the same summation range $q_1 \ldots q_2$. Then if we have $(s', L'_x, L'_y) = \operatorname{argmin}_{s, L_x, L_y} d^{\Sigma}_{w, q_1, q_2}(s, L_x, L_y)$, then we obtain $d^{\Sigma}_{w, q_1, q_2}(s', L'_x, L'_y) \geq d^{\Sigma}_{w, q_1, q_2}(0, L'_x, L'_y)$. Therefore, we will consider only the situation when $s = 0$ and the first nodes of the subpaths $\pi_{x'}$ and $\pi_{y'}$ coincide. A trivial case when we also have $|L_x - L_y| = 0$ is not considered since it is reduced to the absence of a cycle.

Since, in view of Lemma 2, Statement 2 can be applied to the configurations being considered, the condition $|L_x - L_y| \geq \Delta q + 1$ must be satisfied in order that $|L_x - L_y| = \max(s_1, s_2) = \max(s_3, s_4)$ be true. Otherwise, we obtain ed $(x, y) \leq \max(s_1, s_2) + \max(s_3, s_4) < 2(\Delta q + 1)$, contrary to the assumption formulated at the beginning of this proof. From this we obtain $d^{\Sigma}_{w, q_1, q_2}(x, y) \geq \sum_{q = q_1}^{q_2} 2|L_x - L_y| > 2(\Delta q + 1)^2 > (\Delta q + 1)(\Delta q + 2) = Q.$

We combine Lemma 4 with Lemma 3 and formulate Lemma 5.

LEMMA 5. If we assume that $x, y \in \Sigma^w$, $w \geq 6$, and $w \geq q_2 > q_1 > 2w/3$ and that $\Delta q \leq \lfloor (w - q_1 + 1)/2 \rfloor$, $Q \leq 4(w - q_2 + 1)$, and $Q > d^{\Sigma}_{w, q_1, q_2}(x, y)$, then we obtain ed $(x, y) < 2(\Delta q + 1)$.

Proof. It follows from the inequalities $\Delta q \leq \lfloor (w - q_1 + 1)/2 \rfloor$ and $q_1 > 2w/3$ that we have $\Delta q \leq w/6$ and, hence, the values of Δq can be greater than or equal to unity when $w \geq 6$. We use Lemma 3 in the case of (q_1, w)-nonrepetitive strings (case 1) x and y and Lemma 4 in case 2.

We introduce the concept of a good and a bad interval by analogy with the concept of a good or a bad pair of strings. We call a good interval an interval of the form $[i, j]$ and such that, for a pair of strings $x[i, j]$ and $y[i, j]$ bounded by it in x and y, we have $d^{\Sigma}_{w, q_1, q_2}(x[i, j]), y[i, j]) < Q$ (expression (3) under the conditions of Lemma 5).

Next, let us consider intervals of the form $[it' = 1, it' + w]$, where i is a natural number from a definite range, i.e., intervals with a step t'. The lemma formulated below implies that the batched reconstruction of strings containing successive good intervals is possible for $t' < t$, where $t = w - \Delta q + 1$.

LEMMA 6. Let $x, y \in \Sigma^m$, let $\mathbb{N} \ni t'$ divide $(m - w)$, and let $t' < t$. If the conditions of Lemma 5 are fulfilled and, for all $i = 0, \ldots, \frac{m - w}{t'}$, we have $d^{\Sigma}_{w, q_1, q_2}(x[it' + 1, it' + w], y[it' + 1, it' + w]) < Q$, then we obtain ed $(x, y) < 2(\Delta q + 1)$.

Proof. It follows from Lemma 3 that, in the case when coincident q_1-grams are absent for each pair of strings $x' = x[it' + 1, it' + w]$ and $y' = y[it' + 1, it' + w]$ satisfying the condition of Lemma of 5, the paths $\pi_{x'}$ and $\pi_{y'}$ in $B[x, y; q_2]$ form a fork or a shift with a common path fragment consisting of at least $w - q_2 + 1 - 2\Delta q$ edges and we have ed $(x', y') < 2(\Delta q + 1)$. If there are repetitive q_1-grams in $\pi_{x'}$ and/or in $\pi_{y'}$ on the interval $[it' + 1, it' + w - 1]$, then Lemma 4 implies that the substrings $x[it' + 1, it' + w]$ and $y[it' + 1, it' + w]$ can also be interpreted as a shift or a fork (the corresponding paths have a common central fragment of edges, its length equals at least $w - q_2 + 1 - 2\Delta q$, and different edges can be located only at the ends of the interval) and that we also have ed $(x', y') < 2(\Delta q + 1)$.

Let us consider two adjacent intervals $[it' + 1, it' + w]$ and $[(i + 1)t', (i + 1)t' + w]$. Since we have $t' \leq t - 1 = w - \Delta q$, their intersection consists of no less than Δq symbols, which is equal to the maximally possible size of a half-fork of subpaths bounded by good intervals of width w.

The right fork (shift) in the graph $B[x[it' + 1, it' + w], y[it' + 1, it' + w]; q_2]$ and the left fork (shift) in the graph $B[x[(i + 1)t', (i + 1)t' + w], y[(i + 1)t', (i + 1)t' + w]; q_2]$ that belong to the intersection of the intervals consist of the same edges. Therefore, the case is impossible when the mentioned graphs are forks with nonempty half-forks of, respectively, the right and left forks or shifts by different distances or in different directions. Hence, the entire interval $[it' + 1, (i + 1)t' + w]$ is a shift or a fork and the size of the shift or half-fork is also bounded from above by Δq, as well as for the initial intervals. Hence, we have ed $(x[it' + 1, (i + 1)t' + w], y[it' + 1, (i + 1)t' + w]) < 2(\Delta q + 1)$. This reasoning can be extended to other intervals. Then we obtain ed $(x[1, m], y[1, m]) < 2(\Delta q + 1)$.

COROLLARY 1. A sequence (when $t' = 1$) of adjacent bad intervals surrounded by no less than one good interval from each side consists of at least t intervals.

3. A DETERMINISTIC METHOD OF EMBEDDING EDIT DISTANCES INTO l_1

In the deterministic method that is being considered here and is designed for embedding edit distance, an input string $x \in \Sigma^n$ is transformed into a vector $v(x)$ by the concatenation of all the q-gram vectors $v_{i,w,q} = v_{w,q}(x[i, i+w-1])$ for $q = q_1, \ldots, q_2$ and $i = 1, \ldots, n-w+1$. The distance between vectors is assumed to be Manhattan (l_1). In this section, based on the statements obtained in Sec. 1, we find time, resource, and accuracy characteristics of such a deterministic embedding.

We define the following distance for strings $x, y \in \Sigma^n$, using d^{Σ}_{w,q_1,q_2} introduced by formula (1):

$$D(x, y) = \frac{\sum_{i=1}^{n-w+1} d^{\Sigma}_{w,q_1,q_2}(x[i, i+w-1], y[i, i+w-1])}{(n-w+1)(\Delta q+1)} \tag{5}$$

and, with the help of the statements proved in the previous section, show how well the proposed embedding method approximates edit distance.

We will find the lower bound for the number of bad intervals when $\mathrm{ed}(x, y) > k_2$, where k_2 is some parameter of the method. Let the number of bad intervals N be fixed. We first find the upper bound for the cost of editing over all possible arrangements of N intervals, using the algorithm for editing (reconstructing) strings that is presented below.

We pass successively from one interval to other by shifting by one symbol. Assume that we have reconstructed a string up to a position $j-1$. If all the successive intervals beginning with the jth position and ending with the $(j+r)$th one are good, then they are reconstructed with the help of no more than $2(\Delta q+1)$ operations, using the result of Lemma 6 for $t' = 1$. All the intervals at the positions affected by this reconstruction (i.e., between the jth and $(j+r)$th ones) are called covered. If the next jth interval is bad, then we use one edit operation, replace $x[j]$ by $y[j]$ and thereby reconstruct one symbol, and then pass to the next $(j+1)$th interval.

LEMMA 7. Let $S = \left\lfloor \frac{n-1}{w} \right\rfloor$. The reconstruction of a string y from x with the help of the described algorithm requires no more than $2(\Delta q+1)\max(N, 2(\Delta q+1)\times \min(S, N) + \min[N, n-1-(w-1)\min(S, N)]) < 2(\Delta q+1)(N+1)$ edit operations.

Proof. Let us determine the location of bad intervals that maximizes the distance (cost) of editing with the help of the described algorithm. By the definition of edit distance, it will be the upper bound for $\mathrm{ed}(x, y)$.

The maximal cost of editing is attained for the maximal number of occurred situations described in Lemma 6. Therefore, the configuration of bad intervals for which the maximal edit cost is attained assumes the following form of repetitive series: $+ - = = =$, where the signs $+$, $-$, and $=$ denote, respectively, a good interval, a bad interval, and a covered good interval (it is sufficient that only one minus follow a plus to add $2(\Delta q+1)$ operations to the total cost). Moreover, in the sought-for configuration, the last interval will be good since, as a result, $2(\Delta q+1)$ edit operations are added to the total cost.

Thus, the total cost of editing all configurations of the form is $+ - = = =$ amounts to $2(\Delta q+1)\min(S, N)$ (the cost of reconstruction of such configurations that is incremented by their maximal number). At the same time, $n - w\min(S, N) - 1$ bad intervals (unity is subtracted since the last interval always good) or $N - \min(S, N)$ intervals according to which expression, the former or latter, has a smaller value remain unconsidered.

COROLLARY 2. If we have $\mathrm{ed}(x, y) > k_2$ and $q_1 > 2w/3$ and also the conditions of Lemma 5 are satisfied, then we obtain $N > \frac{k_2}{2(\Delta q+1)} - 1$.

Taking into account Corollary 1, one can prove the lemma that is stated below and is similar to Lemma 7.

LEMMA 8. Let the number of bad intervals for strings x and y of length n be fixed, and let it be equal to N. Then we have $\mathrm{ed}(x, y) \le 2(\Delta q+1)\left(\left\lceil \frac{N}{t} \right\rceil + 1\right)$.

COROLLARY 3. If we have ed $(x, y) > k_2$, then we obtain $N > t\left(\frac{k_2}{2(\Delta q + 1)} - 2\right)$.

For $x, y \in \Sigma^w$, we call a q-gram $x[i, i+q-1]$ k-good if a q-gram $y[i, i+q-1]$ can be found such that we obtain $|j - i| \le k$, i.e., a q-gram in the string y shifted by no more than k symbols. If y does not contain such a q-gram, then we call $x[i, i+q-1]$ k-bad. The interval $[i, i+q-1]$ is understood to be k-good if $B[x[i, i+q-1], y[i, i+q-1]; q]$ is a shift and the number of k-good q-grams in both strings is greater than or equal to $w - q + 1 - k$, whence we obtain $d_q(x[i, i+q-1], y[i, i+1-1]) \le 2k$. Otherwise, the interval $[i, i+q-1]$ is understood to be k-bad.

Let $N[R]$ be the number of intervals $[i, i+w-1]$ in which some condition R imposed on these intervals is satisfied.

LEMMA 9. For $x, y \in \Sigma^n$ and $q_1 < q_2 \le w \le \frac{n+1}{k_1+1}$ when ed $(x, y) \le k_1$, we have

$$N[d^{\Sigma}_{w, q_1, q_2}(x[i, i+w-1], y[i, i+w-1]) \le 2k_1(\Delta q + 1)] > n - w(k_1 + 1) + 1.$$

Proof. Let us calculate the minimal number of k_1-good intervals when ed $(x, y) \le k_1$ (see the proof of the Ukkonen lemma [12]). In the aggregate, there are $n - w + 1$ intervals. Since each edit operation can change no more than w intervals, the total number of k_1-bad intervals does not exceed $k_1 w$. The remained $n - w + 1 - k_1 w$ intervals are k_1-good since they can be shifted by no more than k_1 symbols. For a k_1-good interval $[i, i+w-1]$, we have

$$d^{\Sigma}_{w, q_1, q_2}(x[i, i+w-1], y[i, i+w-1]) = \sum_{q=q_1}^{q_2} d_q(x[i, i+w-1], y[i, i+w-1]) < 2k_1(\Delta q + 1).$$

Next, using Corollary 3 and Lemma 9, we can determine the upper and lower bounds for $D(x, y)$ when ed $(x, y) \le k_1$ and ed $(x, y) > k_2$. To this end, we put $w = n^{\gamma}$, $\gamma \le 1 - \frac{\ln(k_1+1)}{\ln n}$.

We denote the set of positions that are the beginnings of bad intervals by $I_B = \{i = 1, \ldots, n-w+1 \mid d^{\Sigma}_{w, q_1, q_2}(x[i, i+w-1], y[i, i+w-1]) \ge Q\}$ and the set of positions that are the beginnings of good intervals by $I_G = \{i = 1, \ldots, n-w+1 \mid d^{\Sigma}_{w, q_1, q_2}(x[i, i+w-1], y[i, i+w-1]) < Q\}$. Then, taking into account Corollary 2, we obtain

$$(n - w + 1)(\Delta q + 1) D(x, y)$$

$$= \left(\sum_{i \in I_B} + \sum_{i \in I_G} \right) \sum_{q=q_1}^{q_2} d_q(x[i, i+w-1], y[i, i+w-1]) \ge NQ + 0 \ge Q\left(\frac{k_2}{2(\Delta q + 1)} - 1\right).$$

Taking into account the grouping of bad intervals (Corollary 3), we obtain

$$D(x, y) \ge \frac{Qt\left(\frac{k_2}{2(\Delta q + 1)} - 2\right)}{(n - w + 1)(\Delta q + 1)} = d_2. \tag{6}$$

We denote the set of positions that are the beginnings of k_1-bad intervals by $I_B^{k_1} = \{i = 1, \ldots, n-w+1 \mid \exists j \in [i - k_1, \ldots, k_1 + i], y[j, j+q-1] = x[i, i+q-1]\}$ and the set of positions that are the beginnings of k_1-good intervals by $I_G^{k_1} = \{i = 1, \ldots, n-w+1 \mid \nexists j \in [i - k_1, \ldots, k_1 + i], y[j, j+q-1] = x[i, i+q-1]\}$. Then, taking into account Lemma 9, we obtain

$$(n - w + 1)(\Delta q + 1) D(x, y) \equiv \left(\sum_{i \in I_B^{k_1}} + \sum_{i \in I_G^{k_1}} \right) \sum_{q=q_1}^{q_2} d_q(x[i, i+w-1], y[i, i+w-1])$$

$$\le (\Delta q + 1)[|I_B^{k_1}|(2w + 2 - q_2 - q_1) + ((n - w + 1) - |I_B^{k_1}|) 2k_1]$$

$$\le 2k_1[w(w + 1 - \frac{q_1 + q_2}{2} - k_1) + (n - w + 1)] < 2k_1[w^2 + (n+1)](\Delta q + 1),$$

whence we have

$$D(x, y) \le \frac{2k_1[w^2 + (n+1)]}{n-w+1} = d_1. \tag{7}$$

The construction of vectors $v(\cdot)$ with the help of a prefix tree requires about $O(n(q_2 + w))$ operations and, accordingly, the estimate of the total construction is $O(n^{1+\gamma})$. When an alphabet is fixed, the dimension of a vector $v(\cdot)$ is determined by the number of levels of the prefix tree Δq and the number of nodes $O(n)$ at each level, and the total dimension amounts to $O(n^{1+\gamma/2})$. The time of computation of the distance between vectors is similarly bounded.

It follows from relationships (6) and (7) that, to ensure the truth of the condition $d_2 > d_1$, it is necessary to have $k_2 = \Omega(k_1(n^{\gamma} + n^{1-\gamma}))$. From this, the optimal value is $\gamma = 1/2$ and, at the same time, we have $k_2 = \Omega(k_1 n^{1/2})$, the time of construction of vectors $v(\cdot)$ equals $O(n^{3/2})$, and their dimension is $O(n^{5/4})$.

4. PROBABILISTIC METHODS OF FORMATION OF VECTOR REPRESENTATIONS FOR SEARCHING FOR NEAREST STRINGS

We consider two randomized versions of the deterministic embedding method described in Sec. 3. These versions can be practically applied to the solution of the nearest-neighbor search problem since the use of its deterministic version is difficult in view of high dimensions of obtained vectors and large sizes of string bases.

Let us consider the nearest-neighbor search (NNS) problem. There are a collection of strings $P = \{p_1, \ldots, p_P \mid p_i \in \Sigma^n\}$ and an input request string $p_0 \in \Sigma^n$; then NNS problem consists of searching for at least one string p^* such that, $\forall\, p \in P$, we have $\text{ed}(p, p_0) \ge \text{ed}(p^*, p_0)$.

For input spaces of high dimensions, the existing exact nearest-neighbor search algorithms are reduced to linear search in P [13], which is frequently unjustified in practice. Moreover, the dimension $O(n^{5/4})$ of the vectors obtained in a deterministic scheme of embedding edit distance into a vector space can turn out to be too high to be efficiently processed. On the other hand, approximate nearest-neighbor search is often sufficient for applications, which has stipulated many publications connected with the development of such algorithms. The *ε-NNS* problem consists of the search for some $p^* \in P$ such that, $\forall p' \in P$, we have $\text{ed}(p^*, p_0) \le (1+\varepsilon)\,\text{ed}(p', p_0)$.

To search for an approximate nearest neighbor, we propose a randomized version of the deterministic algorithm from Sec. 4.1 and also the procedure (from Sec. 4.2) based on locality-sensitive hashing over 1-stable distributions that is relevant to neuro-like distributed representations [9]. Both these versions can be used for solution of the ε-NNS problem.

We give probabilistic variants of Lemma 9 and Corollary 3 for the case when the value of i is randomly selected from a range $1, \ldots, n-w+1$.

COROLLARY 4. If we have $\text{ed}(x, y) \le k_1$ and $q_1 < q_2 \le w \le \dfrac{n+1}{k_1+1}$, then we obtain

$$\text{Prob}\left[d^{\Sigma}_{w, q_1, q_2}(x[i, i+w-1],\ y[i, i+w-1]) \le 2k_1(\Delta q + 1)\right] > 1 - \frac{k_1 w}{n-w+1}.$$

COROLLARY 5. If we have $\text{ed}(x, y) > k_2$, then we obtain

$$\text{Prob}\left[d^{\Sigma}_{w, q_1, q_2}(x[i, i+w-1],\ y[i, i+w-1]) \ge Q\right] > \frac{t\left(\dfrac{k_2}{2(\Delta q+1)} - 2\right)}{n-w+1}.$$

4.1. Solution of the ε-NNS Problem with the Help of Randomization of the Deterministic Method. We use a version of the scheme described in [14]. Let the ith position in a range $1, \ldots, n-w+1$ and the interval $[i, i+w-1]$ corresponding to the range be fixed. If the position i is randomly and equiprobably (with replacement) chosen among all $n-w+1$ possible values, expression (5) is the expectation of the value of $\dfrac{d^{\Sigma}_{w, q_1, q_2}(x[i, i+w-1],\ y[i, i+w-1])}{\Delta q + 1}$. We define a vector $v_i(x)$ as the concatenation of q-gram vectors $v_q(x)$ of this interval for the values of $q = q_1, \ldots, q_2$.

We randomly, independently, and equiprobably (with replacement) choose U values i_f, $f = 1,\ldots,U$, denote their set by I_U, and concatenate $v_{i_f}(x)$ corresponding to them into one vector $\tilde{v}(x)$. We denote the concatenation of all $x[i_f, i_f + w - 1]$ by $x[I_U]$ and put $\tilde{D}(x, y) = \dfrac{||\tilde{v}(x) - \tilde{v}(y)||_{l_1}}{(\Delta q + 1)U}$.

The following lemma is proved by analogy with Lemma 2 [14].

LEMMA 10. Let strings $p_0 \in \sum^n$ and $p_a, p_b \in P$ be such that we have ed $(p_0, p_a) \le k_1$ and ed $(p_0, p_b) > k_2$. Then we obtain

$$\text{Prob}[\tilde{D}(p_0, p_a) > d_1 + \varepsilon] < e^{-2U\varepsilon^2}, \quad \text{Prob}[\tilde{D}(p_0, p_b) < d_2 + \varepsilon] < e^{-2U\varepsilon^2}.$$

We create a structure S consisting of n structures S_k, $k = 1,\ldots,n$, for each possible value of the edit distance between strings of length n. Each structure S_k consists of M structures $F_1,\ldots,F_M$. Each structure F_m, $m = 1,\ldots,M$, contains U positions $i_{m1},\ldots,i_{mU}$ (fixing the beginnings of the corresponding intervals $I_{i_{mu}} = [i_{mu}, i_{mu} + w - 1]$) that are selected randomly and equiprobably (with replacement) from a range $[1,\ldots,n - w + 1]$ and a table Λ_m containing $|\Sigma|^n$ cells (according to the number of possible strings of length n). We denote the content of an interval $I_{i_{mu}}$ in a structure F_m for a string x by $x[I_{i_{mu}}]$. We denote the concatenation of all $v_{w, q_1, q_2}(x[I_{i_{mu}}]), u = 1,\ldots,U$ by $v_m(x)$. The cell that corresponds to $z \in |\Sigma|^{wU}$ in the table Λ_m contains a string $p \in P$ if we have $\tilde{D}(p; z) \le d_1$ and such a p exists; otherwise, the cell is empty. The memory space occupied in the case of the simplest realization of the structure S equals about $O(nM(U + \log P\,|\Sigma|^{wU}) + nP)$ and can be constructed in time $O(nMP(wU + |\Sigma|^{wU}))$.

The size of tables Λ_m can be reduced to $|\Sigma|^{wU}$ by indexing them by strings obtained by concatenating U intervals. Then, to make a decision concerning the assignment of a string to a cell, q-spectra of substrings $z[1 + (u - 1)w, uw]$ are necessary when $q = q_1,\ldots,q_2$. Therefore, it is possible to save memory space for tables Λ_m by uniting the cells with coincident q-spectra of intervals $z[1 + (u - 1)w, uw]$ for all $q = q_1,\ldots,q_2$. In this case, we denote

$$\tilde{D}_m(p; z) = \sum_{u=1}^{U} d^{\Sigma}_{w, q_1, q_2}(p[I_{i_{mu}}], z[1 + (u - 1)w, uw]) = \sum_{u=1}^{U} ||\tilde{v}(p[I_{i_{mu}}]) - \tilde{v}(z[1 + (u - 1)w, uw])||_{l_1}.$$

In the mth table Λ_m, the cell that corresponds to $z \in |\Sigma|^{wU}$ contains a string $p \in P$ if the inequality $\tilde{D}_m(p; z) \le d_1$ is true.

For a request $p_0 \in \Sigma^n$, we consider that a structure F_m makes an error with respect to strings from P if there is a string $p_1 \in P$ such that we have ed $(p_0, p_1) \le k_1$ (or a string p_2 such that we have ed $(p_0, p_2) \ge k_2$) and also $\tilde{D}_m(p_0, p_1) > d_1$ ($\tilde{D}_m(p_0, p_2) < d_2$). If there is some k such that more than $\mu M / \log n$ structures F_m in S_k make errors with respect to strings from P, then we say that the structure S makes errors with respect to the string p_0. We consider that the structure S makes an error with respect to a collection of strings p if there exists $p_0 \in \Sigma^n$ such that S makes an error with respect to the string p_0.

If we assume that we have $M = (n \log_2 \Sigma + \log n - \log \delta) \ln n / \mu$ and $F = 0.5\varepsilon^{-2} \ln(2eP \ln n / \mu)$, then, by Theorem 4 from [14], for any $\delta > 0$ and the request p_0, the probability that S makes an error with respect to p_0 is no more than $\delta 2^{-n}$. Thus, S makes an error with respect to the collection of strings P with probability δ.

Search procedure. Assume that S does not make any error (this occurs with probability $1 - \delta$). We use binary search for finding the nearest neighbor. We choose an arbitrary initial k and randomly and equiprobably select one of structures F_m from S_k. Let us compute the value of $p_0(I_{F_m})$ and check the corresponding cell. If the cell is not empty, then we pass to S_{k-1} and check it. Otherwise, we pass to S_{k+1}. By Lemma 6 from [14], the probability that the selected F_i makes an error with respect to p_0 is less than or equal to μ.

Let us give an analogue of Lemma 7 from [14].

LEMMA 11. If all S_k do not make any error with respect to the string p_0, then the distance from p' returned by the search algorithm differs from the distance from p_0 to the exact nearest neighbor by no more than a factor of $1 + \varepsilon$.

By analogy with Theorem 8 from [14], we obtain that the considered search procedure finds the $(1 + \varepsilon)$-nearest neighbor for any $\varepsilon > 0$ with high probability. If S does not make an error, then the algorithm finds a $(1 + \varepsilon)$-approximate neighbor for any request p_0 with probability $1 - \mu$ in time $O(\varepsilon^{-2} n(\log |P| + \log\log n - \log \mu)\log n)$.

4.2. Solution of the ε-NNS Problem with the Help of the LSH scheme and a 1-Stable Distribution. We first describe an original Locality-Sensitive Hashing (LSH) scheme [15] as applied to strings with the classical edit metric.

Let us define a sphere whose radius is k and that contains points whose distance from the center of the sphere t does not exceed k: $S(t,k)=\{q:\text{ed}(q,s)\le k\}$. The (ε,k)-PLEB (Point Location in Equal Balls) problem is formulated as the search for an algorithm that, for any request $p_0\in P$, performs the following actions: (1) if there exists $p\in P$ such that $p_0\in S(p,k)$, then it returns the answer YES and any of points $p'\in P$ such that $p_0\in S(p',(1+\varepsilon)k)$; (2) if $p_0\notin S(p,(1+\varepsilon)k)$ for all $p\in P$, then it returns the answer NO; (3) if the nearest point $p\in P$ is such that we have $r\le\text{ed}(p_0,p)\le(1+\varepsilon)r$, then it returns the answer YES or NO. In [15], the reducibility of the ε-NNS problem to the ε-PLEB problem is shown. A (k_1,k_2)-PLEB problem is a variant of the ε-PLEB problem and is formulated as the search for an algorithm that returns the answer YES and any of points $p'\in P$ such that we have $p_0\in S(p',k_2)$ if there exists some $p\in P$ such that we have $p_0\in S(p,k_1)$ and, in the other cases, the algorithm returns the answer NO.

The idea of the LSH scheme is as follows: using a definite number of some hash functions, obtain a high probability of collision (the coincidence of values of hash functions) between closely located objects and a low one for distant objects. Then, applying the same hash transformation to a request, we check whether the hash is equal to one of hash values that has been computed earlier for the vectors from P. In the case of such a coincidence, the found vector $p\in P$ is considered to be an approximate nearest neighbor to the request. The magnitude of the distinction of this vector from the exact nearest neighbor depends on the properties of the hash functions being used.

A family $H=\{h:\Sigma^n\to X\}$ (where X is some finite or countable set of values) is called (k_1,k_2,p_1,p_2)-sensitive or simply locality-sensitive if, for any $x, y\in\Sigma^n$ and any independently and equiprobably selected hash function $h\in H$, the following conditions are satisfied:

$$\text{Prob}[h(x)=h(y)]>p_1 \text{ when } x\in S(y,k_1),\qquad \text{Prob}[h(x)=h(y)]<p_2 \text{ when } x\notin S(y,k_2). \tag{8}$$

In order that the family H be "useful," it is necessary that the condition $k_1<k_2$ (i.e., close points must be more close to y than points that are considered to be distant) and also the condition $p_2<p_1$ (close points must cause the collision of hash functions with higher probability than distant ones) be satisfied.

At the preliminary stage of preparation of a structure for approximate nearest-neighbor search, a family of functions $\{g:\Sigma^n\to X^K\}$ is specified, where $g(p)=(h_1(p),h_2(p),\ldots,h_K(p))$. In the aggregate, L functions $g_1,\ldots,g_L$ are randomly and equiprobably selected from the family G. A table is also created and strings p from the base P are assigned to its cell on the basis of the value of the hash vector $g(p)$ as follows: the string p is assigned to the cell corresponding to the value of the hash vector computed from it, namely, $(h_1(p),h_2(p),\ldots,h_K(p))$. The table size is bounded by $O(|P|L)$. Moreover, it is necessary to store the original base, i.e., $O(n|P|)$. The construction of the table is completed when each $p\in P$ is assigned to the corresponding cell.

Search procedure. For a request string p_0, all hash-vectors $g_i(p_0)$, $i=1,\ldots,L$, are computed and the corresponding cells are checked in the table.

If the cell being checked contains a string $p^*\in S(p,k_2)$, then the answer YES and p^* are returned. If, after checking $2L$ strings, none string from P is found in $S(p,k_2)$, then the answer NO is returned.

In [15], the possibility of the truth of the following conditions with a constant probability (that is higher than 1/2) is proved under which the described approximate nearest-neighbor search procedure is correct: (1) if there exists $p^*\in S(p,k_1)$, then the equality $g_j(p)=g_j(p^*)$ must be true for some $j=1,\ldots,L$; (2) the number of collisions of hash functions with strings outside of $S(p,k_2)$ in L checked cells must be less than $2L$, $\sum_{j=1}^{L}|(P-S(q;k_2))\cap g_j^{-1}(g_j(p_0))|<2L$. This is attained by a definite choice of the values of K and L. If H is a (k_1,k_2,p_1,p_2)-sensitive family of functions and we have $K=\log_{1/p_2}|P|$ and $L=|P|^{\rho}$, where $\rho=\ln\left(\frac{p_1}{p_2}\right)$, then the algorithm of solution of the (k_1,k_2)-PLEB problem occupies about $O(n|P|+|P|^{1+\rho})$ memory cells and uses $O(|P|^{\rho})$ computations of distance and $O(|P|^{\rho}\log_{1/p_2}|P|)$ computations of hash functions. It may be noted that $p_2<p_1$ and ρ imply that the search time is sublinearly dependent on $|P|$.

The authors of [16] propose to use the following property of p-stable distributions [17] to construct a family of hash functions that are locality-sensitive to the l_p-norm: linear combinations composed of random quantities ϕ_i are distributed in the same way as one random quantity multiplied by the norm of the coefficients of this linear combination. Since scalar product is linear, the difference $(v_1, \phi) - (v_2, \phi)$ is distributed in the same way as $||v_1 - v_2||_{l_1}\ \phi$. Therefore, if the real axis is divided into equal intervals, then it is intuitively obvious that the scalar products of vectors with close norms by a random vector from the corresponding stable distribution will belong to the same intervals or intervals close to them and, based on this, one can construct a locality-sensitive family.

When $p = 1$ (a 1-stable distribution), to use this property, hash function are specified in the form

$$h(v) = \left\lfloor \frac{(v, \overline{\phi}) + b}{r} \right\rfloor, \tag{9}$$

where $r \in \mathbb{R}$ is some number, b is a uniformly distributed random quantity from $[0, r]$, and $\overline{\phi}$ is a vector of elements from the 1-stable Cauchy distribution with density $\frac{1}{\pi (1 + x^2)}$. It may be proved [16] that, for two fixed vectors v_1 and v_2, depending on the value of the l_1-norm of the difference $c = ||v_1 - v_2||_{l_1}$ of these vectors, the probability of collision of hash function (9) is specified by the expression

$$p(c) = \int_0^r \frac{1}{c} f(c) \left(1 - \frac{t}{c}\right) dt = \frac{1}{\pi} \left(2 \tan^{-1}\left(\frac{r}{c}\right) - \frac{c}{r} \ln\left(1 + \left(\frac{r}{c}\right)^2\right)\right), \tag{10}$$

where $f(\cdot)$ is the density function of the modulus of a Cauchy-distributed random quantity. The function $p(c)$ is monotonically decreasing and, hence, by definition (8), the functions from family (9) are locality-sensitive and they can be used in the LSH scheme.

Next, we find asymptotic values (10) when $r >> c$ and $r << c$.

1. When $r << c$, we have $\tan^{-1}(x) \simeq x - \frac{x^3}{3} + O(x^3)$ and, hence, we obtain

$$p(c) = \frac{1}{\pi} \left(2 \tan^{-1}\left(\frac{r}{c}\right) - \frac{c}{r} \ln\left(1 + \left(\frac{r}{c}\right)^2\right)\right) < \frac{1}{\pi} \left(2 \frac{r}{c} - \ln e^{\frac{r}{c}}\right) = \frac{1}{\pi} \frac{r}{c}\ . \tag{11}$$

2. When $r >> c$ and since we have $\tan(x) \to \frac{\pi}{2}$ as $x \to \infty$, we obtain

$$p(c) = \frac{1}{\pi} \left(2 \tan^{-1} \frac{r}{c} - \frac{c}{r} \ln\left(1 + \left(\frac{r}{c}\right)^2\right)\right) < 1 - \frac{2}{\pi} \frac{c}{r} \ln \frac{r}{c}. \tag{12}$$

We propose a new hash function based on definition (9). In [16], to form a vector $g_j = (h_1(v), h_2(v), \ldots, h_K(v))$, a fixed vector v is scalarwise multiplied by a series of random vectors ϕ_i, $i = 1, \ldots, K$. Instead of such a fixed vector, we take K random vectors $v^i_{w, q_1, q_2}(x)$ that are randomly and independently obtained by the choice of a window of width w in the string x (see Corollaries 5 and 4). For each of them, we generate its vector ϕ_i whose elements are taken from the Cauchy distribution. We define modified hash functions $h'_i(x)$ that are elements of the vector $h'(x) = (h'_1(x), h'_2(x), \ldots, h'_K(x))$ as follows:

$$h'_i(x) = \left\lfloor \frac{(v^i_{w, q_1, q_2}(x), \phi_i) + b}{r} \right\rfloor, \tag{13}$$

where b and r are parameters of formula (9).

To determine the values of p_1 and p_2, it is necessary to find the distribution of quantities $h'_i(x)$ and $h'_i(y)$. Let $\mathrm{Prob}[h'(x) = h'(y) \mid c] = p(c)$, where c is an integral value of $d^{\Sigma}_{w, q_1, q_2}(x, y)$. We denote $d_1 = 2k_1(\Delta q + 1)$, $d_2 = Q$, $p'_1 = 1 - \frac{k_1 w}{n - w + 1}$, and $p'_2 = \frac{t\left(\frac{k_2}{2(\Delta q + 1)} - 2\right)}{n - w + 1}$. Then we have

$$\text{Prob}\,[h_i'(x)=h_i'(y)\,|\,\text{ed}(x,y)\le k_1]\ge \sum_{c\le d_1} p\,(c\,|\,\text{ed}\,(x,y)\le k_1)\,p\,(c)$$

$$\ge p\,(d_1)\sum_{c\le d_1} p\,(c\,|\,\text{ed}\,(x,y)\le k_1)=p\,(d_1)\,p\,(c\le d_1\,|\,\text{ed}\,(x,y)\le k_1)\ge p\,(d_1)\,p_1',$$

$$\text{Prob}\,[h_i'(x)=h_i'(y)\,|\,\text{ed}\,(x,y)>k_2]\le p\,(d_2)\sum_{c\ge d_2} p\,(c\,|\,\text{ed}\,(x,y)>k_2)+\sum_{c<d_2} p\,(c\,|\,\text{ed}\,(x,y)>k_2)\,p\,(c)$$

$$\le p\,(d_2)\,p\,(c\ge d_2\,|\,\text{ed}\,(x,y)>k_2)+\sum_{c<d_2} p\,(c\,|\,\text{ed}\,(x,y)>k_2)\,p\,(0)$$

$$=p\,(d_2)\,p_2'+(1-p_2')\,p\,(0)\le 1-p_2'(1-p\,(d_2)).$$

Thus, for the family of new hash functions (13), we obtain

$$\text{Prob}\,[h_i'(x)=h_i'(y)\,|\,\text{ed}\,(x,y)\le k_1]\ge p\,(d_1)\,p_1'=p_1, \tag{14}$$

$$\text{Prob}\,[h_i'(x)=h_i'(y)\,|\,\text{ed}\,(x,y)>k_2]\le 1-p_2'(1-p\,(d_2))=p_2. \tag{15}$$

In order that this family be locality-sensitive, the condition $1-p_2'(1-p\,(d_2))<p\,(d_1)\,p_1'$ must be true. In turn, in order that this condition be true, the following inequality should be fulfilled:

$$k_2>2(\Delta q+1)(2+\frac{n-w+1}{t\,(1-p\,(d_2))}(1-p\,(d_1)+\frac{wk_1p\,(d_1)}{n-w+1}). \tag{16}$$

To determine the asymptotic behavior of k_2, we put $w=n^{\gamma}$, $0<\gamma<1$, and $r=n^{\mu}$, $\mu>0$. Based on condition (2) of Lemma 3, we have $\Delta q=\Theta(\sqrt{w})$. Then we obtain

$$d_1=2k_1(\Delta q+1)=\Theta(k_1n^{\gamma/2}),\ d_2=Q=\Theta(n^{\gamma}),\ t=w-\Delta q+1=\Theta(n^{\gamma}).$$

As $n\to\infty$, using expressions (10)–(12), we find estimates of $1-p\,(d_1)$ and $1-p\,(d_2)$, depending on the following relationships of the parameters μ and γ.

1. If $\gamma=\mu$, then we have $r=\omega\,(d_1)$ and $r=\Theta\,(d_2)$, whence we obtain

$$1-p\,(d_2)=\text{const},\ 1-p\,(d_1)>\frac{2}{\pi}\frac{d_1}{r}\ln\left(\frac{r}{d_1}\right).$$

2. If $\mu>\gamma$, then we have $r=\omega\,(d_1)$ and $r=\omega\,(d_2)$, whence we obtain

$$1-p\,(d_1)>\frac{2}{\pi}\frac{d_1}{r}\ln\left(\frac{r}{d_1}\right),\ 1-p\,(d_2)>\frac{2}{\pi}\frac{d_2}{r}\ln\left(\frac{r}{d_2}\right).$$

3. If $\gamma/2<\mu<\gamma$, then we have $r=\omega\,(d_1)$ and $r=o\,(d_2)$, whence we obtain

$$1-p\,(d_2)>1-\frac{1}{\pi}\left(\frac{r}{d_2}\right),\ 1-p\,(d_1)>\frac{2}{\pi}\left(\frac{d_1}{r}\ln\left(\frac{r}{d_1}\right)\right).$$

4. If $\mu=\gamma/2$, then we have $r=\omega\,(d_1)$ and $r=o\,(d_2)$, whence we obtain

$$1-p\,(d_2)>1-\frac{1}{\pi}\left(\frac{r}{d_2}\right),\quad 1-p\,(d_1)=\text{const}.$$

Substituting these expressions in inequality (16), we find estimates of k_2 for each case.

1. When we have $\gamma=\mu$, we obtain $k_2=\Omega\,(k_1(n^{1-\gamma}\ln n+n^{\gamma/2})$. When $\gamma>2/3$, $n^{\gamma/2}$ dominates $n^{1-\gamma}\ln n$ and k_2 has the asymptotic $\Omega\,(n^{\gamma/2})$, $\gamma>2/3$. When $\gamma\le 2/3$, $n^{1-\gamma}\ln n$ dominates $n^{\gamma/2}$ and k_2 has the asymptotic $\Omega\,(n^{1-\gamma}\ln n)$, $\gamma\le 2/3$. From this, the optimal value is $\gamma=2/3$ and we have $k_2=\Omega\,(k_1n^{1/3}\ln n)$.

2. When $\mu > \gamma$, we obtain $k_2 = \Omega(n^{\gamma/2} + k_1[n^{1-\gamma} + n^{\mu} / \ln n])$. Since we have $\mu > \gamma$, we obtain $\frac{n^{\mu}}{\ln n} = \Omega(n^{\gamma})$ and, in this case, k_2 increases more quickly than in the case when $\mu = \gamma$.

3. When $\gamma / 2 < \mu < \gamma$, we obtain $k_2 = \Omega(k_1(n^{1-\mu}(\mu - \gamma/2)\ln n + n^{\gamma/2})$. Since we have $\mu < \gamma$, we obtain $n^{1-\mu} \ln n = \Omega(n^{1-\gamma} \ln n)$, and this estimate is worse than in the case when $\mu = \gamma$.

4. When $\mu = \gamma / 2$, we obtain $k_2 = \Omega(n)$, which is also worse than in the case when $n = \gamma$.

Thus, the optimal value is $\gamma = 2/3$ for which we have $k_2^* = \Omega(k_1 n^{1/3} \ln n)$.

Assume that we increase k_2 by equating it to $2(\Delta q + 1)(2 + z\frac{n-w+1}{t(1-p(d_2))}(1-p(d_1)) + \frac{wk_1 p(d_1)}{n-w+1})$ for some $z > 1$. This will affect the value and asymptotic of p_2 since its value depends on k_2. In this case, estimating the value of $\rho = \frac{\ln(p_1' p(d_1))}{\ln(1-p_2'(1-p(d_2)))} = \frac{\ln(1-A)}{\ln(1-zA)}$, where $A = 1 - p_1' p(d_1)$, by analogy with the estimation of ρ for the case of the Hamming distance [15, 18], we obtain $\rho = O\left(\frac{1}{1+z}\right)$ on the condition that the inequality $\ln |P| > p_1' p(d_1)$ is fulfilled.

Thus, the method designed for approximate search for the nearest string and constructed on the basis of the described modified LSH variant over a 1-stable distribution will return a string (from P) that belongs to the sphere $S(q, O(zk_1 n^{1/3} \ln n))$ with probability higher than 1/2 and will use about $O(|P|^{1/(1+z)})$ operations to this end. Since the scheme has a constant probability of error, repeating the described LSH procedure simultaneously and independently about $O(\ln \frac{1}{\alpha})$ times, it is possible to achieve the success probability that is no less than $1 - \alpha$ at least for one procedure start for any given error level $\alpha < 1$.

CONCLUSION

The representation of symbolic strings [8, 9] that is developed within the framework of neural-network-based distributed data representation [19] has been analyzed as applied to the theory of metric embeddings. A new q-gram method of approximation of edit distance is developed and the parameter values of the method are presented for which a refinement of the estimates of approximation quality that are proposed in [9] is attained. An improvement in the asymptotic of growing the lower bound k_2 is achieved in comparison with the results that are obtained using the method of embedding edit distance in the space l_1 and are described in [20]. A randomized version of the approximation method proposed is developed and the possibility of construction of a family of randomized locality-sensitive functions that can be used for efficient solution of the problem of approximate search for nearest strings is demonstrated.

The further directions of investigations are the refinement of the estimates obtained, a modification of the proposed algorithm or the development of a new algorithm for more general versions of edit distance (with permutations and variable costs of operations), and also the investigation of the efficiency of the developed method in practical problems in which long symbolic strings are compared.

REFERENCES

1. V. I. Levenshtein, “Binary Codes Capable of Correcting Deletions, Insertions, and Reversals,” Dokl. Akad. Nauk SSSR, **163**, No. 4, 845–848 (1965).
2. C. Burks, M. J. Cinkosky, and P. Gilna, “Decades of nonlinearity: The growth of DNA sequence data,” in: N. G. Cooper (ed.), Los Alamos Science, No. 20 (1992), pp. 254–255.
3. T. K. Vintsyuk, “Speech recognition by dynamic programming methods,” Cybernetics, No. 1, 81–88 (1968).
4. R. A. Wagner and M. J. Fischer, “The string-to-string correction problem,” J. ACM, **21**, No. 1, 168–173 (1974).
5. G. Navarro, “A guided tour to approximate string matching,” ACM Computing Surveys, **33**, No. 1, 31–88 (2001).
6. P. Indyk, “Embedded stringology,” Talk at Fifteenth Annual Combinatorial Pattern Matching Symposium (2004), http://theory.lcs.mit.edu/~indyk/cpm.ps.

7. P. Indyk, “Open problems,” in: Jiri Matousek (ed.), Workshop on Discrete Metric Spaces and Their Algorithmic Applications, Haifa (2002).
8. A. Sokolov and D. Rachkovskij, “Some approaches to distributed encoding of sequences,” in: Proc. XI-th Intern. Conf. Knowledge–Dialogue–Solution, **2**, Varna, Bulgaria (2005), pp. 522–528.
9. A. Sokolov, “Nearest string by neural-like encoding,” in: Proc. XI-th Intern. Conf. Knowledge-Dialogue-Solution, Varna, Bulgaria (2006), pp. 101–106.
10. N. G. de Bruijn, A Combinatorial Problem, Koninklijke Nederlandsche Akademie van Wetenschappen, **49** (1946).
11. D. Knuth, The Art of Computer Programming, Vol. 2, Seminumerical Algorithms [Russian translation], Mir, Moscow (1977).
12. E. Ukkonen, “Approximate string-matching with *q*-grams and maximal matches,” Theor. Comput. Sci., **92**, No. 1, 191–211 (1992).
13. A. Borodin, R. Ostrovsky, and Y. Rabani, “Lower bounds for high dimensional nearest neighbor search and related problems,” in: Proc. 31st STOC, ACM Press, New York (1999), pp. 312–321.
14. E. Kushilevitz, R. Ostrovsky, and Y. Rabani, “Efficient search for approximate nearest neighbor in high dimensional spaces,” in: Proc. 30th STOC, ACM Press, New York (1998), pp. 614–623.
15. P. Indyk and R. Motwani, “Approximate nearest neighbors: Towards removing the curse of dimensionality,” in: Proc. 30th STOC (1998), pp. 604–613.
16. M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni, “Locality-sensitive hashing scheme based on *p*-stable distributions,” in: 20th Annual Symposium on Computational Geometry, New York (2004), pp. 253–262.
17. J. Nolan, An Introduction to Stable Distributions, http://academic2.american.edu/~jpnolan /stable/chap1.pdf.
18. A. Gionis, P. Indyk, and R. Motwani, “Similarity search in high dimensions via hashing.” in: Proc. VLDB, Morgan Kaufmann Publishers, San Francisco (1999), pp. 518–529.
19. D. A. Rachkovskii, S.V. Slipchenko, E. M. Kussul’, and T. N. Baidyk, “A binding procedure for distributed binary data representations,” Cybernetics and Systems Analysis, No. 3, 3–8 (2005).
20. Z. Bar-Yossef, T. S. Jayram, R. Krauthgamer, and R. Kumar, “Approximating edit distance efficiently,” in: 45th IEEE Symposium on Foundations of Computer Science, IEEE (2004), pp. 550–559.