ГОДИШНИК НА СОФИЙСКИЯ УНИВЕРСИТЕТ „СВ. КЛИМЕНТ ОХРИДСКИ"

ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА
Том 99

ANNUAIRE DE L'UNIVERSITE DE SOFIA „ST. KLIMENT OHRIDSKI"

FACULTE DE MATHEMATIQUES ET INFORMATIQUE
Tome 99

UNIVERSAL LEVENSTEIN AUTOMATA FOR A GENERALIZATION OF THE LEVENSHTEIN DISTANCE

PETAR MITANKIN, STOYAN MIHOV, KLAUS U. SCHULZ

The need to efficiently find approximate matches for a given input string in a large background dictionary arises in many areas of computer science. In earlier work we introduced the concept of a universal Levenshtein automaton for a distance bound n. Given two arbitrary strings v and w, we may use a sequence of bitstrings $\chi(v, w)$ obtained from v and w in a trivial way as input for the automaton. The automaton is deterministic. The sequence $\chi(v, w)$ is accepted iff the Levenshtein distance between v and w does not exceed n. We showed how universal Levenshtein automata can be used to efficiently select approximate matches in large dictionaries. In this paper we consider variants of the Levenshtein distance were substitutions may be blocked for specific symbol pairs. The concept of an universal Levenshtein automaton is extended to cope with this larger class of similarity measures.

1. INTRODUCTION

The problem of how to find good correction candidates for a garbled input word is important for many fundamental applications, including spelling correction, speech recognition, OCR-recognition, error-tolerant querying of search engines for the world wide web and other kinds of information systems. Due to its relevance the problem has been considered by many authors (e.g. [2, 13, 21, 1, 18, 19, 7, 25, 4]).

If an electronic dictionary is available that covers the possible input words, a simple procedure may be used for detecting and correcting errors. Given an input word w, it is first checked if the word is in the dictionary. In the negative case,

the words of the dictionary that are most similar to w are suggested as correction candidates. If necessary, appropriate statistical data can be used for refinement of ranking. Similarity between two words can be measured in several ways. Popular distance measures are the Levenshtein distance ([8, 23, 12, 22, 15, 11]) or n-gram distances ([1, 12, 20, 5, 6]).

In previous research we have shown that for each bound n there exists a finite state automaton - the so-called *universal Levenshtein automaton*, which represents - in some sense - the set of all couples of words $\langle w, v\rangle$ such that the Levenshtein distance between w and v is at most n: Given two arbitrary strings v and w, we may use a sequence of bitstrings $\chi(v, w)$ obtained from v and w in a trivial way as input for the automaton. The sequence $\chi(v, w)$ is accepted iff the Levenshtein distance between v and w does not exceed n. The fact that the automaton is deterministic and does not depend on the particular words but only on the bound n makes the universal Levenshtein automaton very suitable for practical applications. We can use this automaton to extract very efficiently all words from a dictionary that are sufficiently similar to a given input word.

In this paper we show how to compute universal Levenshtein automata for a generalization of the Levenshtein distance. The usual Levenshtein distance represents the minimal number of edit operations required to transform one of the two words into the other. Edit operations are *substitution* (replacement of one symbol of the word with another), *deletion* or *insertion* of a symbol. Here we restrict the set of possible substitutions, thus obtaining a more general and flexible notion of string distance. A set S is fixed that consists of couples of symbols. When we transform one of the two words into the other, i.e. when we calculate the distance, we allow to replace the symbol a with the symbol b only if $\langle a, b\rangle \in S$. When S contains all possible couples, we have the usual Levenshtein distance.

The research on this generalization is motivated by the fact that in many practical applications some of the symbol substitutions are not possible. For instance in a spell checker it would be relevant to restrict S to those couples of symbols whose corresponding keys are situated close to each other on the keyboard or which can have similar phonetic realizations.

The main result presented in this report is a construction of the finite automaton that represents in some sense the set of all couples of words $\langle w, v\rangle$ for which the generalized (via S) Levenshtein distance between w and v is at most n. This automaton has properties analogous to those of the universal Levenshtein automaton.

Ths paper is structured as follows. In Section 2 we start with formal preliminaries. In Section 3 we introduce a non-deterministic variant of the Levenshtein automaton for the generalized distance with restricted substitutions. Section 4 presents a determinization procedure. In Sections 5 and 6we define the corresponding universal automaton. In Section 7 we represent some properties of the universal automaton for the new distance measure. We also present some statistics on the universal automata for bounds $n \leq 5$. The role of each type of automaton will become clearer after reading the formal preliminaries.

2. PRELIMINARIES

Let Σ be finite alphabet and $S \subseteq \Sigma \times \Sigma$. We define d_L^S - function that generalizes the usual Levenshtein distance.

Definition 2.1. $d_L^S : \Sigma^* \times \Sigma^* \to N$
1) $w = \epsilon$ or $x = \epsilon$
$d_L^S(w, x) \stackrel{def}{=} max(|w|, |x|)$
2) $w \neq \epsilon$ and $x \neq \epsilon$
Let $w = w_1 w_2 ... w_p$ and $x = x_1 x_2 ... x_k$.

$$d_L^S(w,x) \stackrel{def}{=} min(\begin{array}{l} if(w_1 = x_1, d_L^S(w_2...w_p, x_2...x_k), \infty), \\ 1 + d_L^S(w_2...w_p, x), \\ 1 + d_L^S(w, x_2...x_k), \\ if(\langle w_1, x_1 \rangle \in S, 1 + d_L^S(w_2...w_p, x_2...x_k), \infty) \quad) \end{array}$$

The following proposition shows that we may think of d_L^S in the terms of *substitution*, *deletion* and *insertion*.

Proposition 2.1. Let us consider that Σ and Σ' are equal but the symbols in Σ are black and the symbols in Σ' are red. In other words, let $\Sigma' = \{a' | a \in \Sigma\}$, r be a biection from Σ into Σ' and $\Sigma \cap \Sigma' = \phi$. For each black symbol $a \in \Sigma$ with a' we denote the corresponding red symbol $r(a)$. Let $v \in (\Sigma \cup \Sigma')^*$. We say that v *is transformed into* w *via deletion of a symbol* iff $v = v_1 v_2 ... v_t$ and $w = v_1 v_2 ... v_{i-1} v_{i+1} ... v_t$ for some i such that $1 \leq i \leq t$ and $v_i \in \Sigma$. We say that v *is transformed into* w *via insertion of a symbol* iff $v = v_1 v_2 ... v_t$ and $w = v_1 v_2 ... v_i b' v_{i+1} ... v_t$ for some i such that $0 \leq i \leq t$ and $b' \in \Sigma'$. We say that v *is transformed into* w *via substitution* iff $v = v_1 v_2 ... v_t$ and $w = v_1 v_2 ... v_{i-1} b' v_{i+1} ... v_t$ for some i such that $1 \leq i \leq t$, $v_i \in \Sigma$, $b' \in \Sigma'$ and $\langle v_i, b \rangle \in S$. If, $w, x \in \Sigma^*$, then $d_L^S(w, x)$ is the minimal natural number k for which there exists sequence of words w_0, w_1, ..., w_k such that
1) $w_0 = w$,
2) if $0 \leq i \leq k-1$ then w_i is transformed into w_{i+1} via deletion of a symbol, insertion of a symbol or substitution,
3) if $w_k = a_1 a_2 ... a_t$ then $x = b_1 b_2 ... b_t$ where $b_i = \begin{cases} a_i & \text{if } a_i \in \Sigma \\ c & \text{if } a_i \in \Sigma' \text{ and } c' = a_i \end{cases}$

In fact this proposition shows that the order of applying the operations that transform w into v is not crucial. For eaxample, if $S = \phi$, then $d_L^S(abc, acd) = 2$. We could apply first the deletion and after it the insertion: $w_0 = abc$, $w_1 = ac$, $w_2 = acd'$. But we could also aplly first the insertion and after it the deletion: $w_0 = abc$, $w_1 = abcd'$, $w_2 = acd'$.

Is it true that d_L^S is a distance? It is true that $d_L^S(w, x) = 0 \Leftrightarrow w = x$ and the triangle inequality holds for d_L^S. But d_L^S is not always symmetric, i.e. d_L^S is not

always distance. d_L^S is distance only when the relation S is symmetric.

How can we compute $d_L^S(w,x)$? Wagner and Fischer show how dynamic programming scheme can be used to compute the usual Levenshtein distance ([23]).The same tehnique can be used for $d_L^S(w,x)$: we find recursively the values M_{ij} of a $(|w|+1)\times(|x|+1)$ sized matrix M:

1) $M_{1j}=j-1$ for $1\leq j\leq |x|+1$ and $M_{i1}=i-1$ for $1\leq i\leq |w|+1$

2) Let us suppose that we have found M_{ij}, $M_{i+1,j}$ and $M_{i,j+1}$. Then

$$\begin{aligned} M_{i+1,j+1} = \; min(\quad & if(w_i=x_j, M_{ij}, \infty), \\ & 1+M_{i,j+1}, \\ & 1+M_{i+1,j}, \\ & if(\langle w_i, x_j\rangle \in S, 1+M_{ij}, \infty) \quad) \end{aligned}$$

After we have found the values of M, we have that $d_L^S(w,x)=M_{|w|+1,|x|+1}$.

Let $n\in N$. We use d_L^S to introduce a criterion for proximity of two words. We consider that the word x is proximate to the word w if $d_L^S(w,x)\leq n$. With $L(w,n)$ we denote the set of all words that are proximate to the word w: $L(w,n)\stackrel{def}{=}\{x|d_L^S(w,x)\leq n\}$. It turns out that for each word w and each n we can build finite automaton $A_n^D(w)$, such that its language $L(A_n^D(w))=L(w,n)$. We give a definition of $A_n^D(w)$ in section 3. The main result of this report is the so-called *universal automaton* $A_n^\forall$ - deterministic finite automaton that represents in some sense $L(w,n)$ for each word w. We call $A_n^\forall$ 'universal' because, in contrast to $A_n^D(w)$, $A_n^\forall$ does not depend neither on particular word w nor on the set S, but it depends only on n. We give a definition of $A_n^\forall$ in section 4.

How does $A_n^\forall$ represent $L(w,n)$ for each word w? Let w be a word and n be a natural number. For given word $x\in\Sigma^+$ we want to know whether $x\in L(w,n)$. We suppose that $|x|\leq |w|+n$. (If $|x|>|w|+n$, then $x\notin L(w,n)$.) $\Sigma_n^\forall$ ($\Sigma_n^\forall$ is the alphabet of $A_n^\forall$) consists of couples of binary vectors, i.e. $\Sigma_n^\forall\subset\{0,1\}^*\times\{0,1\}^*$. By w and x we build in some way a word $\alpha=\alpha_1\alpha_2...\alpha_{|x|}$ whose symbols are couples of binary vectors, i.e. $\alpha_i\in\{0,1\}^*\times\{0,1\}^*$ for $1\leq i\leq |x|$. Then $\alpha\in L(A_n^\forall)\Leftrightarrow x\in L(w,n)$. We build α in the following way: $\alpha=\alpha_1\alpha_2...\alpha_{|x|}$ as $\alpha_i=\langle\beta_i,(\beta_s)_i\rangle$ for $1\leq i\leq |x|$ and

1) $\beta_i=\chi(x_i, w_{i-n}w_{i-n+1}...w_k)$, where

$k=min(|w|,i+n+1)$, $w_{-n+1}=w_{-n}=...=w_0=\$$ for $n>0$, $\$$ is such symbol, that $\$\notin\Sigma$,

and $\chi(c,a_1a_2...a_r)=b_1b_2...b_r$ as $b_j=\begin{cases}1, & \text{if } c=a_j\\ 0, & \text{if } c\neq a_j\end{cases}$

2) $(\beta_s)_i=\chi_s(x_i, w_{i-n+1}w_{i-n+1}...w_k)$, where

$k=min(|w|,i+n-1)$, $w_{-n+2}=w_{-n+1}=...=w_0=\$$ for $n>1$

and $\chi_s(c,a_1a_2...a_r)=b_1b_2...b_r$ as $b_j=\begin{cases}1, & \text{if } \langle a_j,c\rangle\in S\\ 0, & \text{if } \langle a_j,c\rangle\notin S\end{cases}$

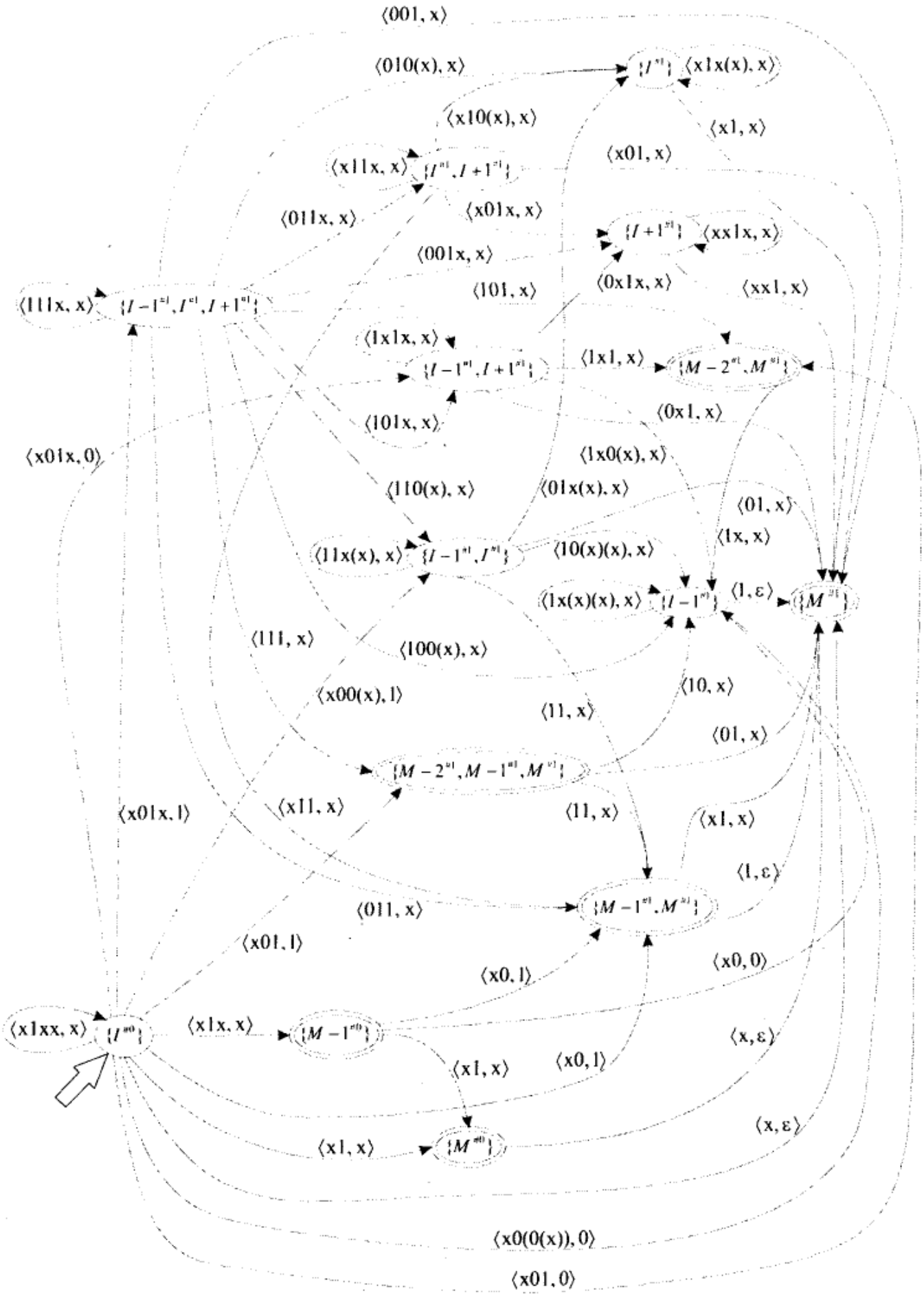

Fig. 1 $A_1^{\forall}$

Example. Let us consider that $\Sigma = \{a, b, c, ..., z\}$ and $S = \{\langle a, d\rangle, \langle d, a\rangle, \langle h, k\rangle, \langle h, n\rangle\}$. Let $w = hahd$ and $x = hand$. We want to know whether $x \in L(hahd, 1)$. We construct the word $\alpha = \alpha_1\alpha_2\alpha_3\alpha_4$:

$\alpha_1 = \langle\beta_1, (\beta_s)_1\rangle = \langle\chi(h, \$hah), \chi_s(h, h)\rangle = \langle 0101, 0\rangle$
$\alpha_2 = \langle\beta_2, (\beta_s)_2\rangle = \langle\chi(a, hahd), \chi_s(a, a)\rangle = \langle 0100, 0\rangle$
$\alpha_3 = \langle\beta_3, (\beta_s)_3\rangle = \langle\chi(n, ahd), \chi_s(n, h)\rangle = \langle 000, 1\rangle$
$\alpha_4 = \langle\beta_4, (\beta_s)_4\rangle = \langle\chi(d, hd), \chi_s(d, d)\rangle = \langle 01, 0\rangle$

The automaton $A_1^{\forall}$ is depicted on fig. 1. On fig. 1 the notation x means 0 or 1 and the bracketed expressions are optional. For instance from the state $\{I - 1^{\#1}, I^{\#1}, I + 1^{\#1}\}$ with $\langle 010(x), x\rangle$ we can reach the state $\{I^{\#1}\}$. This means that from $\{I - 1^{\#1}, I^{\#1}, I + 1^{\#1}\}$ we can reach $\{I^{\#1}\}$ with $\langle 010, 0\rangle$, $\langle 010, 1\rangle$, $\langle 0100, 0\rangle$, $\langle 0100, 1\rangle$, $\langle 0101, 0\rangle$ and $\langle 0101, 1\rangle$. So we start from the initial state $\{I^{\#0}\}$ and with the symbols $\langle 0101, 0\rangle$, $\langle 0100, 0\rangle$, $\langle 000, 1\rangle$ and $\langle 01, 0\rangle$ we visit the states $\{I^{\#0}\}$, $\{I^{\#0}\}$, $\{I - 1^{\#1}, I^{\#1}\}$ and $\{M^{\#1}\}$. $\{M^{\#1}\}$ is final state. Therefore $hand \in L(hahd, 1)$.

With the notions and notations introduced above, the structure of the paper may now be rephrased as follows. In Section we build nondeterministic finite automaton $A_n^{ND}(w)$, such that its language $L(A_n^{ND}(w))$ is $L(w, n)$. In Section we determinize in a specific way $A_n^{ND}(w)$. As a result we obtain the deterministic automaton $A_n^D(w)$. In Section we define the universal automaton $A_n^{\forall}$ and show the connection between $A_n^D(w)$ and $A_n^{\forall}$. In Section we represent some properties of $A_n^{\forall}$, that are based on our previous research. We also show some final results for $A_n^{\forall}$ when $n \leq 5$.

3. NONDETERMINISTIC FINITE AUTOMATON $A_N^{ND}(W)$

Definition 3.1. Let $w \in \Sigma^*$ and $n \in N$.
$A_n^{ND}(w) \stackrel{def}{=} \langle\Sigma, Q_n^{ND}, 0^{\#0}, \delta_n^{ND}, F_n^{ND}\rangle$

Let $|w| = p$. The set of states of $A_n^{ND}(w)$ is $Q_n^{ND} \stackrel{def}{=} \{i^{\#e} | 0 \leq i \leq p \,\&\, 0 \leq e \leq n\}$. (With $i^{\#e}$ we denote the couple $\langle i, e\rangle$.) The set of the final states is $F_n^{ND} \stackrel{def}{=} \{i^{\#e} | p - i \leq n - e\}$. The initial state is $0^{\#0}$. $\delta_n^{ND} \subseteq Q_n^{ND} \times (\Sigma \cup \{\epsilon\}) \times Q_n^{ND}$ is the transition relation. Let $c \in \Sigma \cup \{\epsilon\}$ and $q_1, q_2 \in Q_n^{ND}$. Then

$\langle q_1, c, q_2\rangle \in \delta_n^{ND} \stackrel{def}{\Leftrightarrow}$
$q_1 = i^{\#e} \,\&\, c = w_{i+1} \,\&\, q_2 = i + 1^{\#e}$ or
$q_1 = i^{\#e} \,\&\, c \in \Sigma \,\&\, q_2 = i^{\#e+1}$ or
$q_1 = i^{\#e} \,\&\, c = \epsilon \,\&\, q_2 = i + 1^{\#e+1}$ or
$q_1 = i^{\#e} \,\&\, \langle w_{i+1}, c\rangle \in S \,\&\, q_2 = i + 1^{\#e+1}$

The automaton $A_2^{ND}(w_1w_2w_3w_4w_5)$ is depicted on fig 2. where we use $S_{w_i}^{\epsilon}$ to denote $\{c | \langle w_i, c\rangle \in S\} \cup \{\epsilon\}$.

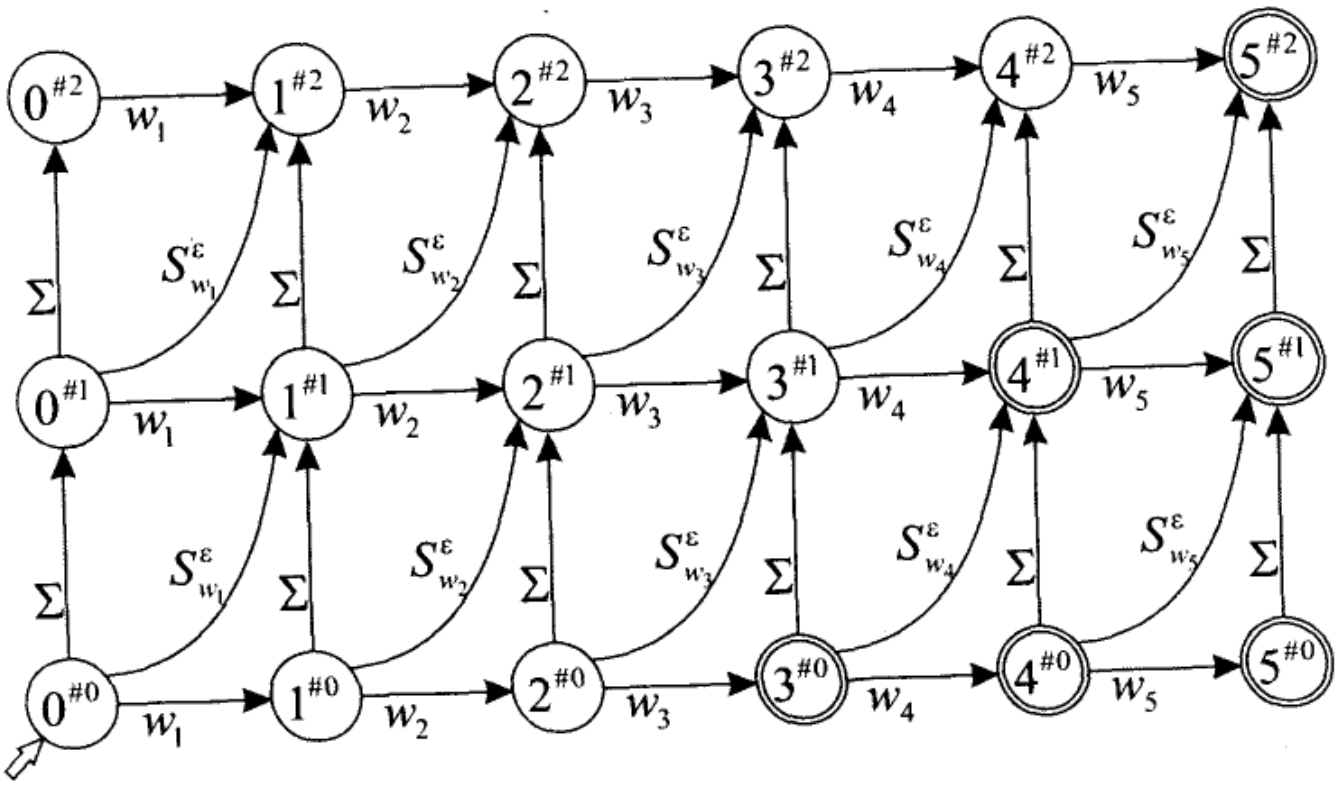

Fig. 2 $A_2^{ND}(w_1w_2w_3w_4w_5)$

If we think of d_L^S in the terms of the operations *substitution*, *deletion* and *insertion*, we can draw the following analogy between $A_n^{ND}(w)$ and d_L^S: each transition $\langle i^{\#e}, \epsilon, i+1^{\#e+1}\rangle$ in $A_n^{ND}(w)$ corresponds to deletion of the symbol w_{i+1}, each transition $\langle i^{\#e}, c, i+1^{\#e+1}\rangle$ for $c \in \Sigma$ corresponds to substitution - replacement of the symbol x_{i+1} with c, each transition $\langle i^{\#e}, c, i^{\#e+1}\rangle$ corresponds to insertion of the symbol c. The e in the state $i^{\#e}$ indicates the number of the operations that have been applied on the way from the initial state $0^{\#0}$ to $i^{\#e}$.

Proposition. $L(A_n^{ND}(w)) = L(w,n)$

Proof. With $\delta_n^{ND^*}$ we denote the extended transition relation, that is defined by induction as usual:

1) $\langle \pi, \epsilon, \pi\rangle \in \delta_n^{ND^*}$

2) $\langle \pi_1, v, \pi'\rangle \in \delta_n^{ND^*}$ & $\langle \pi', a, \pi''\rangle \in \delta_n^{ND}$ & $\langle \pi'', \epsilon, \pi_2\rangle \in \delta_n^{ND^*} \Rightarrow \langle \pi_1, va, \pi_2\rangle$ for $v \in \Sigma^*$ and $a \in \Sigma \cup \{\epsilon\}$

3) $\delta_n^{ND^*}$ is the smallest w. r .t. $\subseteq$ relation for which the conditions 1) and 2) are true.

We check that $L(i^{\#e}) \stackrel{def}{=} \{v \in \Sigma^* | \exists \pi \in F_n^{ND} : \langle i^{\#e}, v, \pi\rangle \in \delta_n^{ND^*}\} = \{v | d_L^S(w_{i+1}w_{i+2}...w_p, v) \leq n-e\} = L(w_{i+1}w_{i+2}...w_p, n-e)$, where $p = |w|$. When $i^{\#e} = 0^{\#0}$, we have $L(A_n^{ND}(w)) = L(0^{\#0}) = L(w,n)$.

Automata that are similar to $A_n^{ND}(w)$ are used for approximate search of a word w in a text T ([24, 3]). If we add a Σ loop to the initial state $0^{\#0}$, the language

of the automaton will be $\Sigma^*.L(w,n)$ and we could use the automaton to traverse the text T.

4. DETERMINISTIC FINITE AUTOMATON $A_N^D(W)$

In this section we determinize the automaton $A_n^{ND}(w)$ in a specific way. In result we get the deterministic automaton $A_n^D(w)$. In the standard subset construction for determinization each state of the deterministic automaton is subset of Q_n^{ND}. We also define each state of $A_n^D(w)$ as a subset of Q_n^{ND}. The difference is that we use the so-called relation of *subsumption* $<_s \subseteq Q_n^{ND} \times Q_n^{ND}$. $<_s$ is defined in such a way that $\pi_1 <_s \pi_2 \Rightarrow L(\pi_1) \supseteq L(\pi_2)$. This allows each state Q of $A_n^D(w)$ to be built such that $\forall \pi', \pi'' \in Q : \pi' \not<_s \pi''$.

The construction that we represent here is analogous to one presented for the usual Levenshtein distance in [17].The main differences are the additional characteristic vector χ_s that depends on S and the additional relevant subword.

4.1. THE RELATION OF SUBSUMPTION $<_S$

Definition 4.1. $Q_n \stackrel{def}{=} \{i^{\#e} | i \in Z \,\&\, 0 \leq e \leq n\}$

Definition 4.2. $<'_s \subseteq Q_n \times Q_n$
$i^{\#e} <'_s j^{\#f} \stackrel{def}{\Leftrightarrow} j^{\#f} \in \{i-1^{\#e+1}, i^{\#e+1}, i+1^{\#e+1}\}$

Proposition 4.1. *Let* $i^{\#e}, j^{\#f} \in Q_n^{ND}$. *Then* $i^{\#e} <'_s j^{\#f} \Rightarrow L(i^{\#e}) \supseteq L(j^{\#f})$.

Definition 4.3. $<_s \subseteq Q_n \times Q_n$
$<_s$ is the transitive closure of $<'_s$.

Corollary 4.1. *Let* $i^{\#e}, j^{\#f} \in Q_n^{ND}$. *Then* $i^{\#e} <_s j^{\#f} \Rightarrow L(i^{\#e}) \supseteq L(j^{\#f})$.

The next proposition gives a direct way to compute whether $i^{\#e} <_s j^{\#f}$.

Proposition 4.2. $i^{\#e} <_s j^{\#f} \Leftrightarrow |j-i| \leq f-e \,\&\, f > e$

The set $\{\pi \in Q_n^{ND} | 3^{\#0} \leq_s \pi\}$ for $n = 2$ and $|w| = 5$ is depicted with bold circles on fig. 3.

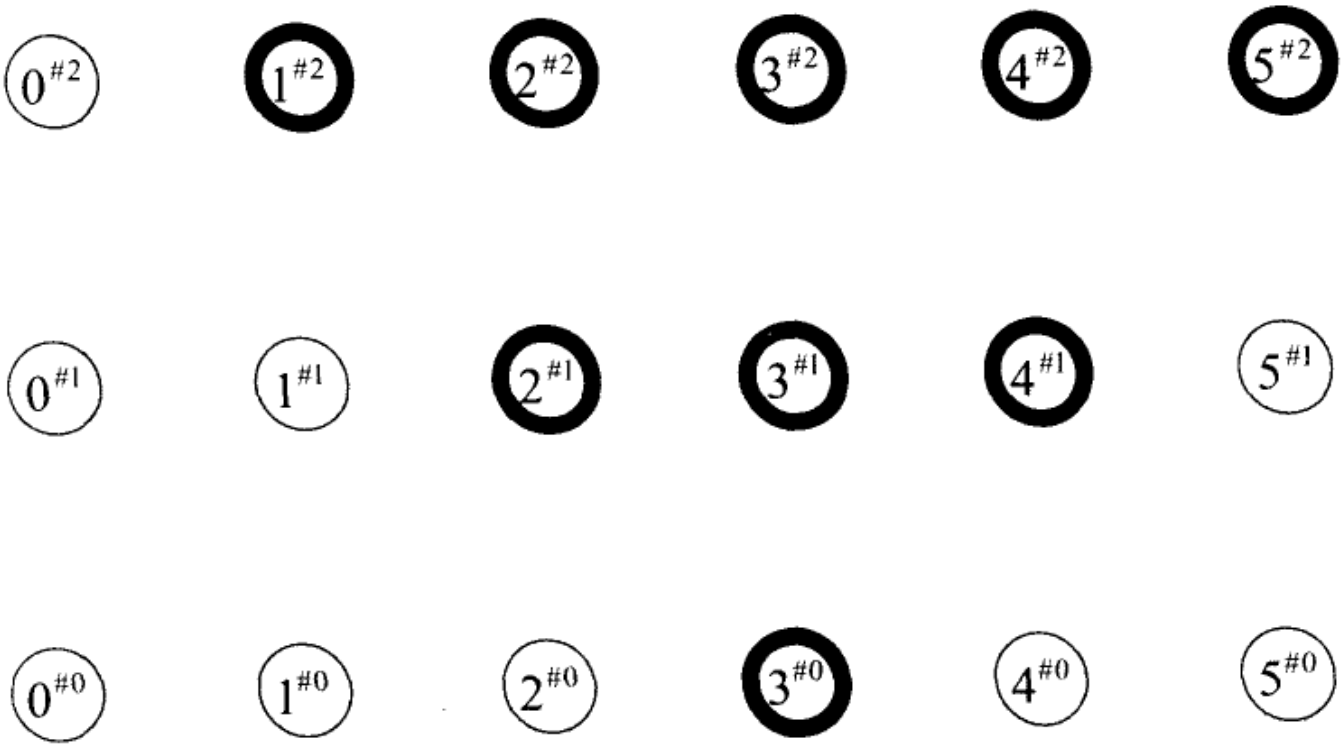

Fig. 3 $\{\pi \in Q_n^{ND} | 3^{\#0} \leq_s \pi\}$

4.2. CHARACTERISTIC VECTORS. RELEVANT SUBWORDS

Let us consider that $A_n^{ND}(w)$ is fixed. Let $\pi \in Q_n^{ND}$ and $a \in \Sigma$.

Definition 4.4. $R(\pi, a) \stackrel{def}{=} \{\pi' \in Q_n^{ND} | \langle \pi, a, \pi' \rangle \in \delta_n^{ND^*}\}$
We call $R(\pi, a)$ *the set of all states reachable from* π *with* a.

To determinize $A_n^{ND}(w)$ we have to know $R(\pi, a)$ for each $\pi \in Q_n^{ND}$ and for each $a \in \Sigma$. We introduce $w_{[\pi]}$ and $w_{[\pi]}^s$ - subwords of w. We call $w_{[\pi]}$ and $w_{[\pi]}^s$ resp. *relevant to* π *subword of* w and *s-relevant to* π *subword of* w. For each symbol $a \in \Sigma$ and each word $a_1 a_2 ... a_k \in \Sigma^*$ we introduce also the binary vectors $\chi(a, a_1 a_2 ... a_k)$ and $\chi_s(a, a_1 a_2 ... a_k)$. We call them resp. *characteristic vector* and *s-characteristic vector of a w. r. t.* $a_1 a_2 ... a_k$. We define the relevant subwords and the characteristic vectors in such a way that if we know $\chi(a, w_{[\pi]})$ and $\chi_s(a, w_{[\pi]})$ then we know $R(\pi, a)$. In $A_n^{ND}(w)$ there are four types of transitions $\langle \tilde{q}_1, c, q_2 \rangle$:

1) $q_1 = i^{\#e}$, $c = w_{i+1}$ and $q_2 = i + 1^{\#e}$
2) $q_1 = i^{\#e}$, $c \in \Sigma$ and $q_2 = i^{\#e+1}$
3) $q_1 = i^{\#e}$, $c = \epsilon$ and $q_2 = i + 1^{\#e+1}$
4) $q_1 = i^{\#e}$, $\langle w_{i+1}, c \rangle \in S$ and $q_2 = i + 1^{\#e+1}$

To know all states $\pi' \in R(\pi, a)$ means to know all sequences
(*) $\langle \pi, \epsilon, \pi_1 \rangle, \langle \pi_1, \epsilon, \pi_2 \rangle, ... \langle \pi_{r-1}, \epsilon, \pi_r \rangle, \langle \pi_r, a, \pi'_1 \rangle, \langle \pi'_1, \epsilon, \pi'_2 \rangle, ... \langle \pi'_h, \epsilon, \pi' \rangle$.

Of course, if we know π then we know all sequences (*), for which the transition $\langle \pi_r, a, \pi'_1 \rangle$ has the type 2). So we define the relevant subwords and the characteristic vectors in such a way that if we know $\chi(\pi, a)$, then we know each sequence (*) for which the transition $\langle \pi_r, a, \pi'_1 \rangle$ has the type 1) or 4):

Definition 4.5. $\chi : \Sigma \times \Sigma^* \to \{0,1\}^*$

$\chi(a, a_1 a_2 ... a_k) = b_1 b_2 ... b_k$ where $b_i = \begin{cases} 1 & \text{if } a = a_i \\ 0 & \text{if } a \neq a_i \end{cases}$

Definition 4.6. $\chi_s : \Sigma \times \Sigma^* \to \{0,1\}^*$

$\chi_s(a, a_1 a_2 ... a_k) = b_1 b_2 ... b_k$ where $b_i = \begin{cases} 1 & \text{if } \langle a_i, a \rangle \in S \\ 0 & \text{if } \langle a_i, a \rangle \notin S \end{cases}$

Definition 4.7. $w_{[\,]} : Q_n^{ND} \to \Sigma^*$

$w_{[i\#e]} \stackrel{def}{=} w_{i+1} w_{i+2} ... w_{i+k}$ where $k = min(n - e + 1, |w| - i)$.

Definition 4.8. $w^s_{[\,]} : Q_n^{ND} \to \Sigma^*$

$w^s_{[i\#e]} \stackrel{def}{=} w_{i+1} w_{i+2} ... w_{i+k}$ where $k = min(n - e, |w| - i)$.

4.3. δ_E^D - THE FUNCTION OF THE ELEMENTARY TRANSITIONS

If we apply the standard subset construction for determinization of $A_n^{ND}(w)$ and A is some state received during the determinization, then $B = \bigcup_{\pi \in A} R(\pi, c)$ will be also state of the deterministic automaton for $c \in \Sigma$. But if $\pi_1, \pi_2 \in B$ and $\pi_1 <_s \pi_2$, we can continue the determinization with $B' = B \backslash \{\pi_2\}$ instead with B, because $\bigcup_{\pi \in B} L(\pi) = \bigcup_{\pi \in B'} L(\pi)$. So we can remove from B each π for which we can find $\pi' \in B$ such that $\pi' <_s \pi$. This means that we can remove from B all states that are not minimal w.r.t. $<_s$. We denote with $\bigsqcup B$ the set of all states that are minimal in B w.r.t. $<_s$. We use also $A \sqcup B$ to denote $\bigsqcup(A \cup B)$ and $\bigsqcup_{\pi \in A} f(\pi)$ to denote $\bigsqcup(\bigcup_{\pi \in A} f(\pi))$.

Definition 4.9. Let $A \subseteq Q_n^{ND}$

$\bigsqcup A \stackrel{def}{=} \{\pi \in A | \neg \exists \pi' \in A(\pi' <_s \pi)\}$

Proposition 4.3. Let $A \subseteq Q_n^{ND}$. Then

$$\bigcup_{q \in A} L(q) = \bigcup_{q \in \bigsqcup A} L(q)$$

We define function $\delta_e^D : Q_n^{ND} \times \Sigma \to P(Q_n^{ND})$, such that $\delta_e^D(\pi, a) = \bigsqcup R(\pi, a)$. The function δ_e^D is called *function of the elementary transitions.*

Definition 4.10. $\delta_e^D : Q_n \times \{0,1\}^* \times \{0,1\}^* \to P(Q_n)$

$\delta_c^D(i^{\#e}, \beta, \beta_s) \stackrel{def}{=} A \sqcup A_S$, where

$$A = \begin{cases} \{i+1^{\#e}\} & \text{if } 1 < \beta \\ \{i^{\#e+1}\} & \text{if } e < n \,\&\, \beta = 0^{k_1} \text{ for some } k_1 \in N \\ \{i^{\#e+1}, i + k_1^{\#e+k_1-1}\}, & \text{if } k_1 \geq 2 \,\&\, 0^{k_1-1}1 < \beta \,\&\, e + k_1 - 1 \leq n \\ \phi & \text{otherwise} \end{cases}$$

$$A_S = \begin{cases} \{i + k_2^{\#e+k_2}\} & \text{if } k_2 > 0 \,\&\, 0^{k_2-1}1 < \beta_s \,\&\, e + k_2 \leq n \\ \phi & \text{ohterwise} \end{cases}$$

Definition 4.11. $\delta_e^D : Q_n^{ND} \times \Sigma \to P(Q_n^{ND})$

$\delta_e^D(\pi, a) \stackrel{def}{=} \delta_c^D(\pi, \chi(a, w_{[\pi]}), \chi_s(a, w_{[\pi]}^s))$

Proposition 4.4. $\bigsqcup R(\pi, a) = \delta_e^D(\pi, a)$

4.4. DETERMINISTIC FINITE AUTOMATON $A_N^D(W)$

Definition 4.12. $A_n^D(w) \stackrel{def}{=} \langle \Sigma, Q_n^D, \{0^{\#0}\}, \delta_n^D, F_n^D \rangle$

The set of states of $A_n^D(w)$ is $Q_n^D \stackrel{def}{=} \{A | A \subseteq Q_n^{ND} \,\&\, \forall \pi_1, \pi_2 \in A(\pi_1 \not<_s \pi_2) \,\&\, \exists i \in [0, |w|] \forall \pi \in A(i^{\#0} \leq_s \pi)\} \backslash \{\phi\}$.

The set of final states is $F_n^D \stackrel{def}{=} \{A \in Q_n^D | A \cap F_n^{ND} \neq \phi\}$.

δ_n^D is partial transition function:

$\delta_n^D : Q_n^D \times \Sigma \to Q_n^D$

1) $\bigcup_{\pi \in A} \delta_e^D(\pi, c) = \phi$

In this case $\delta_n^D(A, a)$ is not defined.

2) $\bigcup_{\pi \in A} \delta_c^D(\pi, a) \neq \phi$

$\delta_n^D(A, c) \stackrel{def}{=} \bigsqcup_{\pi \in A} \delta_e^D(\pi, c)$

The following two propositions give us the correctness of the definition of δ_n^D.

Proposition 4.4. Let $i < |w|$. Then $\forall \pi \in \delta_e^D(i^{\#e}, a) : i + 1^{\#e} \leq \pi$.

Proposition 4.5. $\forall \pi \in \delta_e^D(|w|^{\#e}, a) : |w|^{\#e} \leq \pi$.

From the propositions in 3.3 it follows that $L(A_n^D(w)) = L(A_n^{ND}(w)) = L(w, n)$. The automaton $A_1^D(hahd)$ is depicted on fig. 4. The set S is the one defined in the example in section 1.

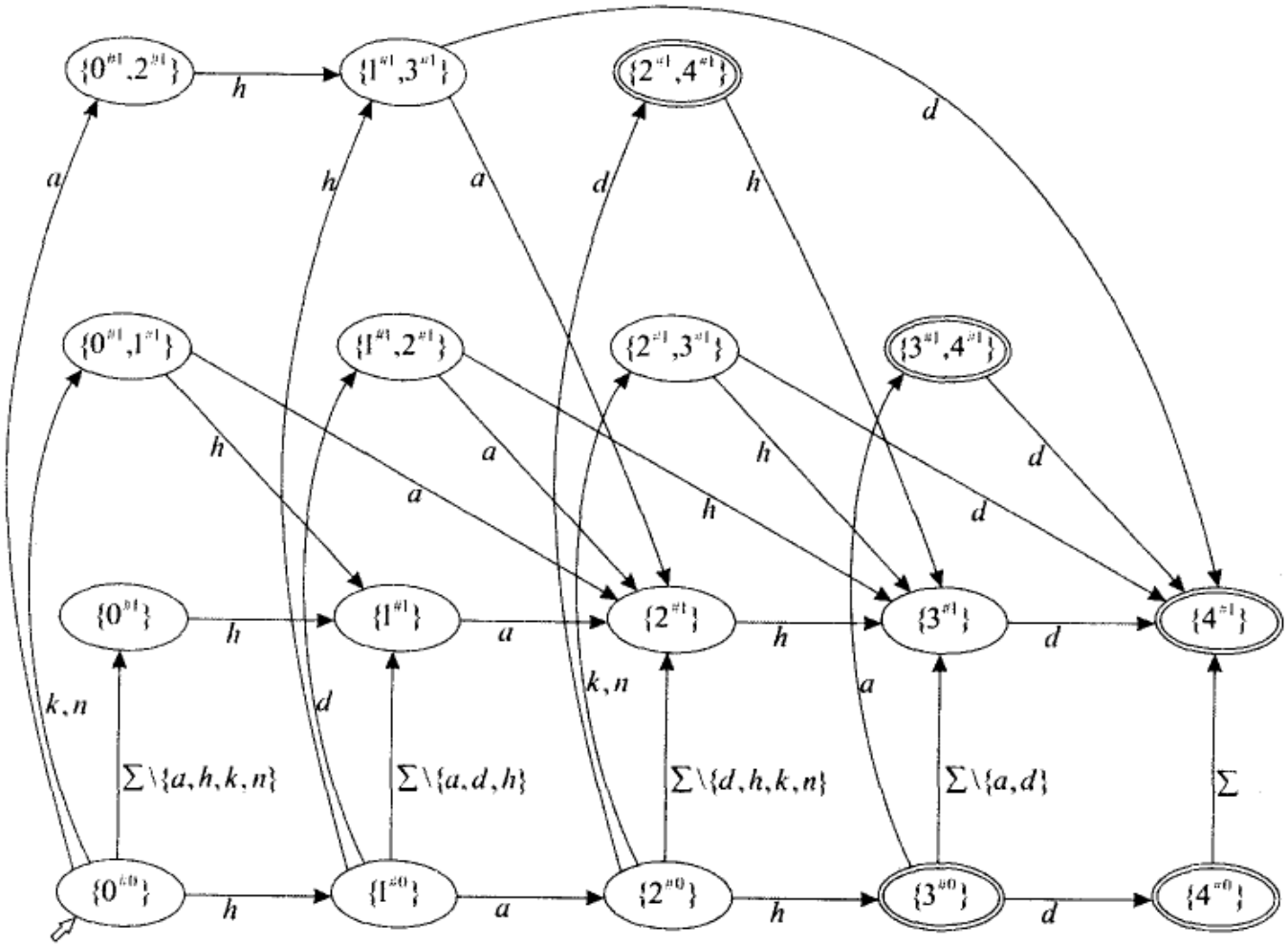

Fig. 4 $A_1^D(hahd)$

5. UNIVERSAL AUTOMATON $A_N^\forall$

We define the universal automaton $A_n^\forall$ in such way that for each execution of $A_n^\forall$ on some word α we can evaluate the execution of $A_n^D(w)$ on x, where w and x are those words from Σ^*, for which we have built the word α in the way described in section 1. The states of $A_n^\forall$ are sets, whose elements have the type $I + a^{\#b}$ or $M + a^{\#b}$ (fig. 1). I and M are parameters. When we replace these parameters with appropriate numbers, the states of $A_n^\forall$ are transformed into the states of $A_n^D(w)$.

Let $q_0^D = \{0^{\#0}\}$, q_1^D,..., q_f^D be those states of $A_n^D(w)$, that we visit with the word $x \in \Sigma^+$. $0 \leq f \leq |x|$. In some cases we may have $f < |x|$ because δ_n^D (the transition function of $A_n^D(w)$) may not be defined. Let $|x| \leq |w| + n$. Let $\alpha = \alpha_1\alpha_2...\alpha_{|x|}$ be built from w and x in the way defined in section 1. Let also $q_0^\forall = \{I^{\#0}\}$, $q_1^\forall$,..., $q_g^\forall$ be those states of $A_n^\forall$, that we visit with the word α. Then:

1) $g = f$,

2) for $0 < i \leq f$ it is true that $q_i^\forall$ is final state iff q_i^D is final state,

3) for $0 \leq i \leq f$ it is true that $q_i^{\forall}$ is transformed into q_i^D by replacement of each I in $q_i^{\forall}$ with i and each M with $|w|$.

Example. Let us look the execution of $A_n^D(w)$ on the word $x = hand$ for the example in section 1. The automaton $A_1^D(w)$ is depicted on fig. 4. So $q_0^D = \{0^{\#0}\}$, $q_1^D = \{1^{\#0}\}$, $q_2^D = \{2^{\#0}\}$, $q_3^D = \{2^{\#1}, 3^{\#1}\}$ and $q_4^D = \{4^{\#1}\}$. In section 1 we saw that $q_0^{\forall} = \{I^{\#0}\}$, $q_1^{\forall} = \{I^{\#0}\}$, $q_2^{\forall} = \{I^{\#0}\}$, $q_3^{\forall} = \{I - 1^{\#1}, I^{\#1}\}$ and $q_4^{\forall} = \{M^{\#1}\}$. If we replace in $q_i^{\forall}$ I with i and M with $|w|$ we get q_i^D.

To the end of section 4 we present the formal definition of the universal automaton $A_n^{\forall} \stackrel{def}{=} \langle \Sigma_n^{\forall}, Q_n^{\forall}, \{I^{\#0}\}, F_n^{\forall}, \delta_n^{\forall} \rangle$.

5.1. $\Sigma_N^{\forall}$, $Q_N^{\forall}$ AND $F_N^{\forall}$

Definition 5.1. $\Sigma_n^{\forall} \subset \{0,1\}^* \times \{0,1\}^*$

$\Sigma_n^{\forall} \stackrel{def}{=} \{\langle \beta, \beta_s \rangle | \beta, \beta_s \in \{0,1\}^* \,\&\, 1 \leq |\beta| \leq 2n+2 \,\&\, 0 \leq |\beta_s| \leq 2n-1\}$

Definition 5.2. $<_s \subseteq Q_n^I \times Q_n^I$

$Q_n^I \stackrel{def}{=} \{I + i^{\#e} | i \in Z \,\&\, 0 \leq e \leq n\}$

$I + i^{\#e} <_s I + j^{\#f} \stackrel{def}{\Leftrightarrow} i^{\#e} <_s j^{\#f}$

Definition 5.3. $<_s \subseteq Q_n^M \times Q_n^M$

$Q_n^M \stackrel{def}{=} \{M + i^{\#e} | i \in Z \,\&\, 0 \leq e \leq n\}$

$M + i^{\#e} <_s M + j^{\#f} \stackrel{def}{\Leftrightarrow} i^{\#e} <_s j^{\#f}$

Definition 5.4. $I_s \subset Q_n^I, M_s \subset Q_n^M$

$I_s \stackrel{def}{=} \{I + i^{\#e} \mid |i| \leq e \,\&\, e \leq n\}$

$M_s \stackrel{def}{=} \{M + i^{\#e} \mid e \geq -i - n \,\&\, i \leq 0 \,\&\, 0 \leq e \leq n\}$

Definition 5.5. $I_{states} \subset P(I_s), M_{states} \subset P(M_s)$

$I_{states} \stackrel{def}{=} \{A | A \subseteq I_s \,\&\, \forall q_1, q_2 \in A(q_1 \not<_s q_2)\} \backslash \{\phi\}$

$M_{states} \stackrel{def}{=} \{A | A \subseteq M_s \,\&\, \forall q_1, q_2 \in A(q_1 \not<_s q_2) \,\&$
$\exists j \exists f (M + j^{\#f} \in A \,\&\, M + j^{\#f} \leq_s M^{\#n}) \,\&\, \exists i \in [-n, 0] \forall q \in A(M + i^{\#0} \leq_s q)\}$

$Q_n^{\forall} \stackrel{def}{=} I_{states} \cup M_{states}$

$F_n^{\forall} \stackrel{def}{=} M_{states}$

6.1. $\delta_E^{\forall}$ - THE FUNCTION OF THE ELEMENTARY TRANSITIONS

Definition 6.1. $r_n : (I_s \cup M_s) \times \{0,1\}^* \to \{0,1\}^*$

1) $r_n(I + i^{\#e}, x_1x_2...x_k) \stackrel{def}{=} \begin{cases} x_{n+i+1}x_{n+i+2}...x_{n+i+h} & \text{if } h \geq 0 \\ \text{not defined} & \text{if } h < 0 \end{cases}$

where $h = min(n - e + 1, k - n - i)$.

2) $r_n(M+i^{\#e}, x_1x_2...x_k) \stackrel{def}{=} \begin{cases} x_{k+i+1}x_{k+i+2}...x_{k+i+h} & \text{if } h \geq 0 \,\&\, k+i+1 > 0 \\ \text{not defined} & \text{otherwise} \end{cases}$

where $h = min(n - e + 1, -i)$.

Definition 6.2. $r_n^s : (I_s \cup M_s) \times \{0,1\}^* \to \{0,1\}^*$

1) $r_n^s(I + i^{\#e}, x_1x_2...x_k) \stackrel{def}{=} \begin{cases} x_{n+i}x_{n+i+1}...x_{n+i+h-1} & \text{if } h \geq 0 \,\&\, n+i > 0 \\ \epsilon & \text{otherwise} \end{cases}$

where $h = min(n - e, k - n - i + 1)$.

2) $r_n^s(M+i^{\#e}, x_1x_2...x_k) \stackrel{def}{=} \begin{cases} x_{k+i+1}x_{k+i+2}...x_{k+i+h} & \text{if } h \geq 0 \,\&\, k+i+1 > 0 \\ \epsilon & \text{otherwise} \end{cases}$

where $h = min(n - e, -i)$.

Definition 6.3. $I : P(Q_n) \to P(Q_n^I)$

$I(A) \stackrel{def}{=} \{I + i - 1^{\#e} | i^{\#e} \in A\}$

Definition 6.4. $M : P(Q_n) \to P(Q_n^M)$

$M(A) \stackrel{def}{=} \{M + i^{\#e} | i^{\#e} \in A\}$

Definition 6.5. $\delta_e^{\forall} : (I_s \cup M_s) \times \Sigma_n^{\forall} \to P(I_s) \cup P(M_s)$

Let $A \in I_s \cup M_s$ and $\langle \beta, \beta_s \rangle \in \Sigma_n^{\forall}$.

1) $r_n(A, \beta)$ is not defined

In this case $\delta_e^{\forall}(A, \langle \beta, \beta_s \rangle)$ is not defined.

2) $r_n(A, \beta)$ is defined

2.1) $A \in I_s$

Let $q = i^{\#e}$ where i and e are such that $A = I + i^{\#e}$.

$\delta_e^{\forall}(A, \langle \beta, \beta_s \rangle) \stackrel{def}{=} I(\delta_e^D(q, r_n(A, \beta), r_n^s(A, \beta_s)))$

2.2) $A \in M_s$

Let $q = i^{\#e}$ where i and e are such that $A = M + i^{\#e}$.

$\delta_e^{\forall}(A, \langle \beta, \beta_s \rangle) \stackrel{def}{=} M(\delta_e^D(q, r_n(A, \beta), r_n^s(A, \beta_s)))$

Definition 6.6. $rm : I_{states} \cup M_{states} \to I_s \cup M_s$
Let $A \in I_{states} \cup M_{states}$.
Let $t = \begin{cases} \mu z[\ \exists e \exists i(z = e - i \ \& \ I + i^{\#e} \in A & \text{if } A \in I_{states} \\ \mu z[\ \exists e \exists i(z = e - i \ \& \ M + i^{\#e} \in A & \text{if } A \in M_{states} \end{cases}$
With $\mu z[X]$ we denote the least z such that the X is true.
$rm(A) \overset{def}{=} \begin{cases} I + i^{\#e} & \text{if } I + i^{\#e} \in A \ \& \ e - i = t \\ M + i^{\#e} & \text{if } M + i^{\#e} \in A \ \& \ e - i = t \end{cases}$

$rm(A)$ is called *right most element of A.*

Definition 6.7. $\bigtriangledown_a : I_{states} \cup M_{states} \to P(N)$
1) $A = \{I^{\#0}\}$
$\bigtriangledown_a(A) \overset{def}{=} \{k | n \leq k \leq 2n + 2\}$
2) $A \in I_{states}$ & $A \neq \{I^{\#0}\}$
Let $rm(A) = I + i^{\#e}$.
$\bigtriangledown_a(A) \overset{def}{=} \{k | 2n + i - e + 1 \leq k \leq 2n + 2\}$
3) $A \in M_{states}$
$\bigtriangledown_a(A) \overset{def}{=} \{k \in N | \forall q \in A(if(k < n, M^{\#n-k}, M + n - k^{\#0}) \leq_s q)\} \backslash \{0\}$

Definition 6.8. $l_n : N \times N \to \{true, false\}$
$l_n(k_1, k_2) = true \overset{def}{\Leftrightarrow} (k_1 = 2n + 2 \ \& \ k_2 = 2n - 1)$ or $(k_1 = 2n + 1 \ \& \ k_2 = 2n - 1)$ or $(1 \leq k_1 \leq 2n \ \& \ k_2 = k_1 - 1)$

6.3. SOME OTHER FUNCTIONS AND $\delta_N^{\forall}$

Definition 6.9. $f_n : (I_s \cup M_s) \times N \to \{true, false\}$
1) $f_n(I + i^{\#e}, k) \overset{def}{=} \begin{cases} true & \text{if } k \leq 2n + 1 \ \& \ e \leq i + 2n + 1 - k \\ false & \text{otherwise} \end{cases}$
2) $f_n(M + i^{\#e}, k) \overset{def}{=} \begin{cases} true & \text{if } e > i + n \\ false & \text{otherwise} \end{cases}$

Definition 6.10. $m_n : (Q_n^I \cup Q_n^M) \times N \to Q_n^I \cup Q_n^M$
$m_n(A, k) \overset{def}{=} \begin{cases} M + i + n + 1 - k^{\#e} & \text{if } A = I + i^{\#e} \\ I + i - n - 1 + k^{\#e} & \text{if } A = M + i^{\#e} \end{cases}$
$m_n : (P(Q_n^I) \cup P(Q_n^M)) \times N \to P(Q_n^I) \cup P(Q_n^M)$
$m_n(A, k) \overset{def}{=} \{m_n(a, k) | a \in A\}$

Definition 6.11. $\bigsqcup : P(P(I_s)) \cup P(P(M_s)) \to P(I_s) \cup P(M_s)$
$\bigsqcup A \overset{def}{=} \{q \in \cup A | \neg \exists q' \in \cup A : q' <_s q\}$

Definition 6.12. $\delta_n^{\forall} : Q_n^{\forall} \times \Sigma_n^{\forall} \to Q_n^{\forall}$

Let $A \in Q_n^\forall$ and $\langle \beta, \beta_s \rangle \in \Sigma_n^\forall$.
1) $|\beta| \notin \bigtriangledown_a(A)$ or $l_n(|\beta|, |\beta_s|) = false$
In this case $\delta_n^\forall(A, \langle \beta, \beta_{subs} \rangle)$ is not defined.
2) $|\beta| \in \bigtriangledown_a(A)$ & $l_n(|\beta|, |\beta_s|) = true$
2.1) $\bigcup_{q \in A} \delta_e^\forall(q, \langle \beta, \beta_{subs} \rangle) = \phi$
In this case $\delta_n^\forall(A, \langle \beta, \beta_{subs} \rangle)$ is not defined.
2.2) $\bigcup_{q \in A} \delta_e^\forall(q, \langle \beta, \beta_{subs} \rangle) \neq \phi$
Let $\Delta = \bigsqcup_{q \in A} \delta_e^\forall(q, \langle \beta, \beta_{subs} \rangle)$.

$$\delta_n^\forall(A, \langle \beta, \beta_{subs} \rangle) \stackrel{def}{=} \begin{cases} \Delta & \text{if } f_n(rm(\Delta), |\beta|) = false \\ m_n(\Delta, |\beta|) & \text{if } f_n(rm(\Delta), |\beta|) = true \end{cases}$$

7. SOME PROPERTIES OF $A_N^\forall$

When $S = \Sigma \times \Sigma$ d_L^S is the usual Levenshtein distance that we denote with d_L. In [27] and [26] we have shown that for d_L one can build universal automaton which here we denote with $A_n^u = \langle \Sigma_n^u, Q_n^u, \{I^{\#0}\}, \delta_n^u, F_n^u \rangle$. In this section we show the connection between $A_n^\forall$ and A_n^u and some corollaries.

Σ_n^u is the set of the first projections of the elements of $\Sigma_n^\forall$, i.e. $\Sigma_n^u = \{\beta \in \{0,1\}^* | 1 \leq |\beta| \leq 2n+2\}$. To define A_n^u we use the sets I_s and M_s, the relation $<_s$ and the sets I_{states} and M_{states} defined in section 4. So each state in Q_n^u, just like each state in $Q_n^\forall$, is a subset of I_s or subset of M_s. In [26] we have shown the following:

1) for A_n^u it is true that if $q \in I_{states} \cup M_{states}$, then q is useful in the sense that q is reachable from the initial state $\{I^{\#0}\}$ and some final state is reachable from q,

2) A_n^u is minimal.

Proposition 7.1 (for the connection between A_n^u and $A_n^\forall$). *Let $q \in Q_n^u$ and $\beta = \Sigma_n^u$. Then $\exists! \beta_s \in \{1\}^* : \delta_n^u(q, \beta) = \delta_n^\forall(q, \langle \beta, \beta_s \rangle)$ (either both the left expression and the right expression are not defined or both the left expression and the right expression are defined and equal).*

Remark. That β_s, for which $\delta_n^u(q, \beta) = \delta_n^\forall(q, \langle \beta, \beta_s \rangle)$, is $\beta_s = 1^{k_2}$ where k_2 is such that $l_n(|\beta|, k_2) = true$.

It follows from the proposition for the connection that 1) and 2) hold also for $A_n^\forall$. $Q_n^u = Q_n^\forall$. In [26] we have presented rough upper limitation for $|Q_n^u|$:
$|I_{states}| = O(2^{4n - \log_2 \sqrt{2n+1}})$
$|M_{states}| = O(n 2^{4n - \log_2 \sqrt{2n+1}})$
In the table below we show some final results for $A_n^\forall$ when $n \leq 5$. The value of the column 'transitions' is $|\{\langle q_1, b, q_2 \rangle | \langle q_1, b, q_2 \rangle \in \delta_n^\forall\}|$.

Table 1.

n	$\lvert I_{states}\rvert$	$\lvert M_{states}\rvert$	transitions
1	8	6	320
2	50	40	39552
3	322	280	4480416
4	2187	2025	504895904
5	15510	15026	58028259232

8. CONCLUSION

Besides the operations insertion, deletion and substitution in many applications there is a need for adding other operations. For example for spell checker it would be relevant to add *transposition* (swap two adjacent symbols) - mistake that occurs very frequently while typing text on keyboard. To correct text recognized by an OCR program it would be useful to add *merge* (merge of two adjacent symbols into one) and *split* (split one symbol into two others). In [17] and [26] we have shown that in the case of adding transposition as well as in the case of adding merge and split we can build deterministic automaton and universal automaton such that the universal one simulates the deterministic one: The technique developed in this research can be successfully applied if we restrict the allowed operations in these cases. For instance restricting the allowed substitutions, the allowed merges and the allowed splits results in universal automaton whose alphabet consists of fourtuples of binary vectors: besides χ and χ_s we add two other characteristic vectors that depend on the allowed merges and the allowed splits.

Here comes the problem for characterization of all functions $d : \Sigma^* \times \Sigma^* \to N$ for which universal automaton can be built. Our future research will be devoted to this problem.

REFERENCES

1. Angell, R., George E. Freund, and Peter Willett. Automatic spelling correction using a trigram similarity measure. *15 Information Processing and Management*, **19**, 255–261, 1983.
2. Blair, C. A program for correcting spelling errors. *Information and Control*, **3**, 60–67, 1960.
3. Baeza-Yates R., Gonzalo Navarro. Faster approximate string matching. *Algorithmica*, **23**, 2, 127–158, 1999.
4. Dengel A., Rainer Hoch, Frank Hones, Thorsten Jager, Michael Malburg, and Achim Weigel. Techniques for improving OCR results. In: H. Bunke and P. S. P.Wang, editors, Handbook of Character Recognition and Document Image Analysis. World Scientific, 1997.

5. Kim, Jong Yong, John Shawe-Taylor. An approximate string-matching algorithm. *Theoretical Computer Science*, **92**, 107–117, 1992.
6. Kim Jong Yong, John Shawe-Taylor. Fast string matching using an n-gram algorithm. *Software–Practice and Experience*, **94**, 1, 79–88, 1994.
7. Kukich, K. Techniques for automatically correcting words in texts. *ACM Computing Surveys*, **24**, 377–439, 1992.
8. Levenshtein, V. Binary codes capable of correcting deletions, insertions, and reversals. *Sov. Phys. Dokl.*, **10**, 07– 710, 1966.
9. Mitankin, P. Universal Levenshtein automata - building and properties. Technical report, FMI, University of Sofia, 2005. Master thesis.
10. Stoyan Mihov and Klaus U. Schulz. Fast approximate search in large dictionaries. *Computational Linguistics*, **30**, 4, 451–477, December 2004.
11. B. John Oommen and Richard K.S. Loke. Pattern recognition of strings with substitutions, insertions, deletions, and generalized transpositions. *Pattern Recognition*, **30**, 5, 789–800, 1997.
12. Olumide Owolabi and D.R. McGregor. Fast approximate string matching. *Software - Practice and Experience*, **18**, 4, 387–393, 1988.
13. Edward M. Riseman and Roger W. Ehrich. Contextual word recognition using binary digrams. *IEEE Transactions on Computers*, **C-20**, 4, 397–403, 1971.
14. Sargur N. Srihari, Jonathan J. Hull, and Ramesh Choudhari. Integrating diverse knowledge sources in text recognition. *ACM Transactions on Information Systems*, **1**, 1, 68–87, 1983.
15. Giovanni Seni, V. Kripasundar, and Rohini K. Srihari. Generalizing edit distance to incorporate domain information: Handwritten text recognition as a case study. *Pattern Recognition*, **29**, 3, 405–414, 1996. 16
16. Klaus U. Schulz and Stoyan Mihov. Fast String Correction with Levenshtein-Automata. Technical Report Report 01-127, CIS University of Munich, 2001.
17. Klaus U. Schulz and Stoyan Mihov. Fast String Correction with Levenshtein-Automata. *International Journal of Document Analysis and Recognition*, **5**, 1, 67–85, 2002.
18. Sargur N. Srihari. Computer Text Recognition and Error Correction. Tutorial, IEEE Computer Society Press, Silver Spring, MD, 1985.
19. Hiroyasu Takahashi, Nobuyasu Itoh, Tomio Amano, and Akio Yamashita. A spelling correction method and its application to an OCR system. *Pattern Recognition*, **23**, 3/4, 363–377, 1990.
20. Esko Ukkonen. Approximate string-matching with q-grams and maximal matches. *Theoretical Computer Science*, **92**, 191– 211, 1992.
21. Je.rey R. Ullmann. A binary n-gram technique for automatic correction of substitution, deletion, insertion and reversal errors. *The Computer Journal*, **20**, 2, 141–147, 1977.
22. Achim Weigel, Stephan Baumann, and J. Rohrschneider. Lexical postprocessing by heuristic search and automatic determination of the edit costs. In: Proc. of the Third International Conference on Document Analysis and Recognition (ICDAR 95), pages 857–860, 1995.

23. Robert A.Wagner and Michael J. Fischer. The string-to-string correction problem. *Journal of the ACM*, **21**, 1, 168–173, 1974.

24. Sun Wu and Udi Manber. Fast text searching allowing errors. *Communications of the ACM*, **35**, 10, 83–91, 1992.

25. Justin Zobel and Philip Dart. Finding approximate matches in large lexicons. *Software–Practice and Experience*, **25**, 3, 331– 345, 1995.

26. Petar Mitankin. Universal Levenshtein Automata - Building and Properties, FMI, University of Sofia, 2005, Master thesis.

27. Stoyan Mihov and Klaus U. Schulz, Fast Approximate Search in Large Dictionaries, *Computational Linguistics*, 2004, **30**, 4, 451-477.

Received on September 26, 2006

Faculty of Mathematics and Informatics
"St. Kl. Ohridski" University of Sofia
5, J. Bourchier blvd., 1164 Sofia
BULGARIA
E-mail: peromit@yahoo.com