Fuzzy String Matching with Finite Automat

Armen Kostanyan
IT Educational and Research Center of
Yerevan State University
Yerevan, Armenia
armko@ysu.am

***Abstract*—The string matching problem is one of the widely-known symbolic computation problems having applications in many areas of artificial intelligence. The most famous algorithms solving the string matching problem are the finite automata method, Knuth-Morris-Pratt and Rabin-Karp's algorithms. In this paper we focus on applying the finite automata method to find a fuzzy pattern in a text.**

***Keywords*— string matching with finite automata, fuzzy sets, fuzzy string matching.**

I. Introduction

The string matching (i.e., the problem of finding all of the locations of a specific pattern in a string) is a well-known computational problem the earliest references to which date to the 1960s. The interest to this problem really took off with the publication of Boyer-Moor (BM) [1] and Knuth-Morris-Pratt (KMP) [2] algorithms in 1970s. A number of exact string matching algorithms have been proposed since then using one or another modification of either the BM or KMP.

Along with investigations on exact string matching, there have also been investigations on approximate string matching in which the problem of finding a generic pattern was studied. By contrast to the ordinary pattern specified as a list of characters, the generic pattern is rather a structural description of a substring to be found. The investigations on approximate string matching were addressed to the distance based string matching [3], the string matching with the use of patterns with meta-characters and, more generally, with the use of patterns represented by regular expressions [4, 5]. Comprehensive review of these works was done in monograph [6].

In this paper we formulate the problem of fuzzy string matching where the pattern is represented as a sequence of term-values of a *linguistic variable* [7] and the problem of finding all of the occurrences of such a pattern in a text with a given accuracy is defined. This problem is investigated with the use of the finite automata in the following way. A non-deterministic transition system is constructed to describe the opportunities of processing the given text with the purpose of finding all of the occurrences of a fuzzy pattern in it, whereupon an efficient algorithm to determine some of the occurrences is proposed.

The paper is organized as follows.

Section 2 presents the finite automata method for exact string matching. Section 3 provides basic concepts of fuzzy set theory and formulates the problem to be investigated. Section 4 introduces the transition system to describe the string matching opportunities. An example of applying this system is provided in Section 5. Section 6 presents a greedy algorithm for partially solving the fuzzy string matching problem. Finally, the conclusion summarizes the obtained results.

II. String Matching with a Finite Automaton

The classical string matching problem is formulated as follows.

We are given a text $T[1..n]$ of length n and a pattern $P[1..m]$ of length m ($n \geq m$). It is assumed that the elements of P and T are characters drawn from a finite alphabet Σ. We say that pattern P *occurs with shift* s in text T if $0 \leq s \leq n-m$ and $T[s+1..s+m]=P[1..m]$ (that is, $T[s+j]=P[j]$ for $1 \leq j \leq m$). If P occurs with shift s in T, then s is said to be a *valid shift*; else it is said to be an *invalid shift*. The *string-matching problem* is the problem of finding all valid shifts with which P occurs in T.

The finite automata method is based on the suffix function which is defined as follows.

Given a pattern $P[1..m]$ over an alphabet Σ, the *suffix function* for P is defined as a mapping σ: $\Sigma^* \rightarrow \{0, 1, \dots, m\}$ such that

$$\sigma(x)=\max\{\, k \mid 0 \leq k \leq m,\ P_k \text{ is a suffix of } x \,\},$$

where P_k denotes $P[1..k]$.

The pattern $P[1..m]$ derives a finite automaton $M_P=(Q, q_0, F, \Sigma, \delta)$ such that

- $Q=\{ 0, 1, \dots, m \}$ is the set of states;
- $q_0=0$ is the initial state;
- $F=\{ m \}$ is the one-element set of final states;
- δ: $Q \times \Sigma \rightarrow Q$ is the transition function such that $\delta(q, a)=\sigma(P_q a)$ for all $q \in Q$ and $a \in \Sigma$.

Claim: Pattern $P[1..m]$ occurs with shift s in text $T[1..n] \Leftrightarrow M_P$ accepts the string $T[1..s+m]$.

Once the automaton M_P is constructed, all valid shifts of P in text T can be determined in $O(n)$ time. Taking into account

This work was supported by the RA MES State Committee of Science, in the frames of the research project N 15T-18350.

978-1-5386-2830-0/17/$31.00 ©2017 IEEE

that with the use of the *prefix function* (which is another string matching function) M_P can be constructed in O($|\Sigma| \cdot m$) time, we get the O($n+|\Sigma| \cdot m$) total time for string matching with a finite automaton.

III. The Fuzzy String Matching Problem

Let us generalize the classical string matching problem by formulating the problem of finding a *fuzzy pattern* in a text.

Suppose $(L, \vee, \wedge, 0, 1)$ is a finite lattice with the least element 0 and the greatest element 1. According to [7], a *fuzzy subset A* of a universal set U is defined by a membership function μ_A: $U \rightarrow L$ that associates with each element x of U a number $\mu_A(x)$ in L representing the *grade of membership* of x in A. A fuzzy subset A of U can be represented as an additive form

$$A = \sum_{x \in U} x / \mu_A(x).$$

We say that an element x definitely belongs to A, if $\mu_A(x)=1$, and it definitely does not belong to A, if $\mu_A(x)=0$. In contrast, if $0<\mu_A(x)<1$, we say that x belongs to A with degree $\mu_A(x)$.

Let us define a *fuzzy symbol t* over the alphabet Σ to be a fuzzy subset of Σ. Given a character $a \in \Sigma$ we say that a matches t with grade $\mu_t(a)$.

Given a set Ξ of fuzzy symbols, we define the *fuzzy pattern* $P[1..m]$ to be a sequence of symbols from Ξ of length m. Given a threshold $\lambda \in L$, we say that a pattern $P[1..m]$ λ-occurs in a text $T[1..n]$ with shift s, if $T[s+j]$ matches $P[j]$ with grade at least λ for all j, $1 \le j \le m$. We say that s is a λ-valid shift, if P λ-occurs in T with shift s. Finally, let us define the *λ-fuzzy string matching problem* to be the problem of finding all λ-valid shifts of the fuzzy pattern P in text T.

IV. Processing Text by a Transition System

Suppose the automaton M_P with transition function δ_P has been constructed for a pattern $P[1..m]$ over the alphabet Ξ. We shall describe the solution to the λ-fuzzy string matching problem in terms of a nondeterministic transition system the states of which are pairs $s=<q, \alpha>$, where $0 \le q \le m$, α is a sequence of L-values of length q. We interpret the state $s=<q, <\alpha_1, \ldots, \alpha_q>>$ in the following way: if $T[h+1], \ldots, T[h+q]$ is the sequence of the last q read characters, then

$$\alpha_i = \mu_{P[i]}(T[h+i]), \quad 1 \le i \le q.$$

More precisely, given $\lambda \in L$, consider the transition system $W_P=(S, s_0, \Phi, \Sigma, \Delta_P)$, where

- $S=\{ s=<q, \alpha>| \ 0 \le q \le m, \alpha=<\alpha_1, \ldots, \alpha_q>, \alpha_i \in L$ for all $1 \le i \le q \}$;
- $s_0=<0, <>>$ is the start state;
- $\Phi=\{ s=<m, <\alpha_1, \ldots, \alpha_m>> \mid \alpha_i \ge \lambda , 1 \le i \le m \}$ is the set of final states;
- Σ is the text alphabet;
- Δ_P: $S \times \Sigma \rightarrow 2^S$ is the transition function such that

$$[q < m, q \xrightarrow{t} (q+1) \in \delta_P] \Rightarrow$$

$$< q, < \alpha_1, \ldots \alpha_q > \xrightarrow{a} < q+1, < \alpha_1, \ldots \alpha_q, \mu_t(a) >> \in \Delta_P;$$

$$[q \xrightarrow{t} q' \in \delta_P, q' \le q] \quad \Rightarrow$$

$$< q, < \alpha_1, \ldots \alpha_q >> \xrightarrow{a} < q', < \alpha_{q-q'+2}, \ldots \alpha_q, \mu_t(a) >> \in \Delta_P,$$

for all $\alpha_1, \ldots, \alpha_q \in L$, $a \in \Sigma$;

There are no other transitions in Δ_P.

Suppose ω: $\Sigma^* \rightarrow 2^S$ is the final-state function for W_P such that

$\omega(\varepsilon)=\{ s_0 \}$,

$\omega(xa)=\{ s' \mid$ there exists $s \in \omega(x)$ such that $s' \in \Delta_P(s, a)\}$.

Theorem 1: $\omega(T_{s+m}) \cap \Phi \ne \varnothing \Leftrightarrow s$ is a λ-valid shift.

V. Example

Let us choose $\Sigma=\{ 1, 2, 3, 4, 5 \}$, $L=\{ 0, 0.25, 0.5, 0.75, 1 \}$ and define SMALL and LARGE fuzzy symbols as follows:

SMALL= 1/1 + 2/0.75 + 3/0.5 + 4/0.25 + 5/0,
LARGE= 1/0 + 2/0.25 + 3/0.5 + 4/0.75 + 5/1.

Assume that Ξ={SMALL, LARGE}, P= SMALL.LARGE.SMALL (here "." is used as a separator of symbols from Ξ). Fig. 1 presents the diagram of the automaton M_P:

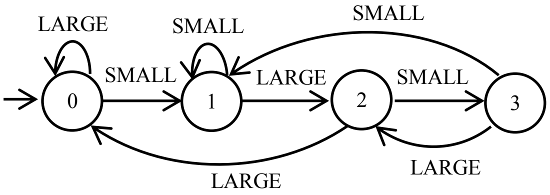

Fig. 1. The finite automaton derived from
P=SMALL.LARGE.SMALL.

Assuming that T=32415231, let us consider the following processing of T by the transition system S:

$$\begin{array}{lll}
s_0 =< 0, <>> & \xrightarrow{3} & s_1 =< 0, <>> \\
 & \xrightarrow{2} & s_2 =< 1, < 0.75 >> \\
 & \xrightarrow{4} & s_3 =< 2, < 0.75, 0.75 >> \\
 & \xrightarrow{1} & s_4 =< 3, < 0.75, 0.75, 1 >^* \\
 & \xrightarrow{5} & s_5 =< 2, < 1, 1 >> \\
 & \xrightarrow{2} & s_6 =< 3, < 1, 1, 0.75 >>^* \\
 & \xrightarrow{3} & s_7 =< 2, < 0.75, 0.5 >> \\
 & \xrightarrow{1} & s_8 =< 3, < 0.75, 0.5, 1 >> .
\end{array}$$

It follows from this execution of T that at λ=0.75 we have two final states (that is, s_4 and s_6) and, respectively, two 0.75-valid shifts (that is, 1 and 3).

At λ=0.5 we would have three final states (that is, s_4, s_6 and s_8) and three 0.5-valid shifts (that is, 1, 3 and 5).

VI. Partial Fuzzy String Matching

The transition system constructed in Section 3 can be restricted in one or another way to construct an approximate algorithm that finds some of the occurrences of a fuzzy pattern in a given text. One of such approximate algorithms is provided below in which the transition of the automaton M_P most suitable for next symbol of the given text is always chosen.

In the description of the algorithm we shall use the *singled-valued* function $\gamma_{qq'}$: $L^q\times\Sigma\rightarrow L^{q'}$ defined for all $0\leq q$, $q'\leq m$ in the following way:

$$\gamma_{qq'}(\alpha, a)=\alpha' \Leftrightarrow <q', \alpha'>\in\Delta_P(<q, \alpha>, a)$$
$$\text{for all } \alpha\in L^q \text{ and } a\in\Sigma.$$

Let us denote by $pr_1(s)$ and $pr_2(s)$ the first and second components of the state $s\in S$, respectively. Finally, for $\alpha=<\alpha_1, ..., \alpha_k>$ denote by $\min(\alpha)$ the least component of α, i. e., $\alpha_1 \wedge \ldots \wedge \alpha_m$.

Algorithm. PARTIAL FUZZY STRING MATCHER.
Input: Fuzzy pattern $P[1..m]$, text $T[1..n]$, threshold λ $(0<\lambda\leq1)$.
Method:

```
currState=<0,<>>
for i = 1 to n
    maxGrade=0
    for all $t\in\Xi$
        if $\mu_t(T[i])>maxGrade$
            $maxGrade=\mu_t(T[i])$
            sym=t
            $currState=<q', \gamma_{qq'}(pr_2(currState), T[i])>$,
                where $q= pr_1(currState)$, $q'=\delta_P(q, sym)$
    if $pr_1(currState)==m$ && $\min(pr_2(currState))\geq\lambda$
        Print ("Pattern $\lambda$-occurs with shift", i – m)
```

This algorithm recognizes all 0.75-valid shifts from the example in Section 4. On the other hand, to recognize the 0.5-valid shift 5, the algorithm must perform the LARGE-transition among two transitions with the same rate 0.5 while reading the second character 3 from state 3.

The complexity of the algorithm is $O(n\cdot(|\Xi|+m))$. Considering the $O(m\cdot|\Xi|)$ time needed for the construction of the automaton M_p, we get $O(n\cdot(|\Xi|+m))+O(m\cdot|\Xi|)= O(n\cdot(|\Xi|+m))$ total time for partial fuzzy string matching.

VII. Conclusion

The problem of finding occurrences of a fuzzy pattern in a given text with a given accuracy has been considered in this paper. A nondeterministic transition system is constructed to describe the set of all possible ways of processing the pattern reading the text. This transition system is restricted to obtain a $O(n\cdot(|\Xi|+m))$-time algorithm for finding some of the occurrences of a fuzzy pattern in the given text.

References

[1] R. S. Boyer and J. S. Moore, "A fast string matching algorithm," *Association for Computing Machinery,* vol. 20, no. 10, pp. 762-772, 1977.

[2] D. Knuth and M. Pratt, "Fast pattern matching in strings," *SIAM J. Comput.* vol. 6, no. 2, pp. 323-350, 1977.

[3] G. Landau and U. Vishkin, "Efficient string matching with k mismatches," *TCS,* vol. 43, pp. 239-249, 1986.

[4] R. Baeza-Yates and N. Gonzaio, "A faster algorithm for approximate string matching," in *Proc. of Seventh Annual Symp. Combinatorial Pattern Matching*, Spinger-Verlag, 1996, pp.1-23.

[5] R. Baeza-Yates and N. Gonzaio, "Multiple approximate string matching," in *Proc. of Fifth Annual Workshop on Algorithms & Data Structures*, 1997, pp. 174-184.

[6] B. Smyth, *Computing Patterns in Strings*, Addison-Wesley UK, 2003.

[7] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning-I," *Information Sciences,* vol. 8, pp. 199-249, 1975.