Journal of Universal Computer Science, vol. 1, no. 5 (1995), 292-311
submitted: 21/1/95, accepted: 14/5/95, appeared: 28/5/95, ©Springer Pub. Co.

Lexical Analysis with a Simple Finite-Fuzzy-Automaton Model [1]

Alexandru Mateescu
Academy of Finland and the Mathematics Department, University of Turku,
20500 Turku, Finland.

Arto Salomaa
Academy of Finland and the Mathematics Department, University of Turku,
20500 Turku, Finland.

Kai Salomaa
Mathematics Department, University of Turku, 20500 Turku, Finland.

Sheng Yu
Department of Computer Science, The University of Western Ontario, London,
Ontario, Canada N6A 5B7

Abstract: Many fuzzy automaton models have been introduced in the past. Here, we discuss two basic finite fuzzy automaton models, the Mealy and Moore types, for lexical analysis. We show that there is a remarkable difference between the two types. We consider that the latter is a suitable model for implementing lexical analysers. Various properties of fuzzy regular languages are reviewed and studied. A fuzzy lexical analyzer generator (FLEX) is proposed.

Category: G.2

1 Introduction

In most of the currently available compilers and operating systems, input strings are treated as crisp tokens. A string is either a token or a non-token; there is no middle ground. For example in UNIX, if you enter "yac", it does not mean "yacc" to the system. If you type "spelll" (the key sticks), it will also not be treated as "spell" although there is no confusion. Would it be more friendly if the

[1]The work reported here has been supported by the Natural Sciences and Engineering Research Council of Canada grants OGP0041630 and the Project 11281 of the Academy of Finland.

system would ask you whether you meant "yacc" in the first case and "spell" in the second case, or simply decide for you if there is no confusion ? Sure. There are many different ways available which can be used to implement the above idea. Various models of fuzzy automata have been introduced in, e.g., [7], [10], [6], [2], [11]. However, what is needed here is a model that is so simple, so easy to implement, and so efficient to run that it makes sense to be utilized.

Here we describe a very simple model for this purpose. The fuzzy automaton model we describe in this article follows those described in [7], [4], [5], [10], [2], [11], etc. in principle. Fuzzy languages and grammars were formally defined by Lee and Zadeh in [4]. Maximin automata as a special class of pseudo automata were studied by Santos [9, 10]. A more restricted, Mealy type model was also studied by Mizumoto et al. [5]. Note that many concepts described in this paper are not new. The main purpose of this article is to try to renew an interest in fuzzy automata as language acceptors and, especially, their applications in lexical analysis and parsing.

In the following, we first review the basic concept of fuzzy languages and define regular fuzzy languages. Then we describe two types of finite fuzzy automata, the Mealy and Moore types. We compare them and argue that one is better than the other for the purpose of lexical analysis. We also study other properties of finite fuzzy automata and their relation to fuzzy grammars [4].

2 Regular fuzzy languages

Many of the following basic definitions on fuzzy languages can be found in [4].

Definition 2.1 *Let Σ be a finite alphabet and $f : \Sigma^* \to M$ a function, where M is a set of real numbers in $[0, 1]$. Then we call the set*

$$\tilde{L} = \{(w, f(w)) \mid w \in \Sigma^*\}$$

a fuzzy language over Σ and f the membership function of $\tilde{L}$.

In the following, we often use $f_{\tilde{L}}$ to denote the membership function of $\tilde{L}$.

Let $\tilde{L}$ be a fuzzy language over Σ and $f_{\tilde{L}} : \Sigma^* \to M$ the membership function of $\tilde{L}$. Then, for each $m \in M$, denote by $S_{\tilde{L}}(m)$ the set

$$S_{\tilde{L}}(m) = \{w \in \Sigma \mid f_{\tilde{L}}(w) = m\}.$$

Note that $S_{\tilde{L}}$ as a function is just $f_{\tilde{L}}^{-1}$.

Definition 2.2 *Let* $\tilde{L}_1$ *and* $\tilde{L}_2$ *be two fuzzy languages over* Σ. *Then the basic operations on* $\tilde{L}_1$ *and* $\tilde{L}_2$ *are define in the following:*

(1) The membership function $f_{\tilde{L}}$ *of the union* $\tilde{L} = \tilde{L}_1 \cup \tilde{L}_2$ *is defined by*

$$f_{\tilde{L}}(w) = max\{f_{\tilde{L}_1}(w), f_{\tilde{L}_2}(w)\},\ w \in \Sigma^*.$$

(2) The membership function $f_{\tilde{L}}$ *of the intersection* $\tilde{L} = \tilde{L}_1 \cap \tilde{L}_2$ *is defined by*

$$f_{\tilde{L}}(w) = min\{f_{\tilde{L}_1}(w), f_{\tilde{L}_2}(w)\},\ w \in \Sigma^*.$$

(3) The membership function $f_{\tilde{L}}$ *of the complement of* $\tilde{L}_1$ *is defined by*

$$f_{\tilde{L}}(w) = 1 - f_{\tilde{L}_1}(w),\ w \in \Sigma^*.$$

(4) The membership function $f_{\tilde{L}}$ *of the concatenation* $\tilde{L} = \tilde{L}_1 \cdot \tilde{L}_2$ *is defined by*

$$f_{\tilde{L}}(w) = max\{min(f_{\tilde{L}_1}(x), f_{\tilde{L}_2}(y)) \mid w = xy,\ x, y \in \Sigma^*\},\ w \in \Sigma^*.$$

(5) The membership function $f_{\tilde{L}}$ *of the star operation* $\tilde{L} = \tilde{L}_1^{\,*}$ *is defined by*

$$f_{\tilde{L}}(w) = max\{min(f_{\tilde{L}_1}(x_1), \ldots, f_{\tilde{L}_1}(x_n)) \mid w = x_1 \cdots x_n,$$
$$x_1, \ldots, x_n \in \Sigma^*, n \geq 0\},\ w \in \Sigma^*,$$

assuming that $min\emptyset = 1$.

(6) The membership function $f_{\tilde{L}}$ *of the + operation* $\tilde{L} = \tilde{L}_1^{\,+}$ *is defined by*

$$f_{\tilde{L}}(w) = max\{min(f_{\tilde{L}_1}(x_1), \ldots, f_{\tilde{L}_1}(x_n)) \mid w = x_1 \cdots x_n,$$
$$x_1, \ldots, x_n \in \Sigma^*, n \geq 1\},\ w \in \Sigma^*.$$

Since fuzzy languages are just a special class of fuzzy sets, the equivalence and inclusion relations between two fuzzy languages are the equivalence and equivalence relations between two fuzzy sets. Let $\tilde{L}_1$ and $\tilde{L}_2$ be two fuzzy languages over Σ. Then

$$\tilde{L}_1 = \tilde{L}_2 \text{ iff } f_{\tilde{L}_1}(w) = f_{\tilde{L}_2}(w) \text{ for all } w \in \Sigma^*,$$

and

$$\tilde{L}_1 \subseteq \tilde{L}_2 \text{ iff } f_{\tilde{L}_1}(w) \leq f_{\tilde{L}_2}(w) \text{ for all } w \in \Sigma^*.$$

Definition 2.3 *Let $\tilde{L}$ be a fuzzy language over Σ and $f_{\tilde{L}} : \Sigma^* \to M$ the membership function of $\tilde{L}$. We call $\tilde{L}$ a regular fuzzy language if*

(1) the set $\{m \in M \mid S_{\tilde{L}}(m) \neq \emptyset\}$ is finite and

(2) for each $m \in M, S_{\tilde{L}}(m)$ is regular.

It is obvious that the first condition can be replaced by

(1') M is finite.

For convenience, when we write $f_{\tilde{L}} : \Sigma^* \to M$, we mean that $M = \{f_{\tilde{L}}(w) \mid w \in \Sigma^*\}$, i.e., for each $m \in M, S_{\tilde{L}}(m) \neq \emptyset$. Also, the second condition in the above definition can be replaced by

(2') for each $m \in M$, $\{w \in \Sigma^* \mid f_{\tilde{L}}(w) \geq m \}$ is regular.

We choose (2) instead of (2') since it can be used more directly in the subsequent proofs.

Example 2.1 *Let $\tilde{L}_1$ be a fuzzy language over $\Sigma = \{a, b\}$ and $f_{\tilde{L}_1}$:*

$$f_{\tilde{L}_1}(x) = \begin{cases} 1, & \text{if } x \in a^*, \\ 0.7, & \text{if } x \in a^*ba^*, \\ 0.5, & \text{if } x \in a^*ba^*ba^*, \\ 0, & \text{otherwise.} \end{cases}$$

Then $\tilde{L}_1$ is a regular fuzzy language.

Example 2.2 *The membership function $f_{\tilde{L}_2}$ of $\tilde{L}_2$ over $\Sigma = \{a, b\}$ is defined by*

$$f_{\tilde{L}_2}(x) = |x|_a/|x|,$$

where $|x|$ denotes the length of x and $|x|_a$ the number of appearances of a in x. Then $\tilde{L}_2$ is not a regular fuzzy language.

The next theorem can be easily proved.

Theorem 2.1 *Regular fuzzy languages are closed under union, intersection, complement, concatenation, and star operations.*

Proof. Let $\tilde{L}_1$ and $\tilde{L}_2$ be two regular fuzzy languages over Σ and $f_{\tilde{L}_1} : \Sigma^* \to M_1$ and $f_{\tilde{L}_2} : \Sigma^* \to M_2$. Let $\tilde{L}$ be the resulting language after an operation (union, ..., or star) and $f_{\tilde{L}} : \Sigma^* \to M$. Then obviously, $M \subseteq M_1 \cup M_2$ (in the case of a union, an intersection, a concatenation, or a star operation) or $M = \{1 - m \mid m \in M_1\}$ (in the case of a complementation) is finite. Let m be in M. Then $S_{\tilde{L}}(m)$ is defined by:

(1) Union:

$$S_{\underset{\sim}{L}}(m) = \begin{cases} S_{\underset{\sim}{L_1}}(m) - \bigcup_{m'>m} S_{\underset{\sim}{L_2}}(m'), & \text{if } m \in M_1 - M_2, \\ S_{\underset{\sim}{L_2}}(m) - \bigcup_{m'>m} S_{\underset{\sim}{L_1}}(m'), & \text{if } m \in M_2 - M_1, \\ ((S_{\underset{\sim}{L_1}}(m) \bigcup S_{\underset{\sim}{L_2}}(m)) - \bigcup_{m'>m} S_{\underset{\sim}{L_1}}(m')) - \bigcup_{m''>m} S_{\underset{\sim}{L_2}}(m''), & \text{if } m \in M_1 \cap M_2; \end{cases}$$

(2) Intersection:

$$S_{\underset{\sim}{L}}(m) = \begin{cases} S_{\underset{\sim}{L_1}}(m) - \bigcup_{m'<m} S_{\underset{\sim}{L_2}}(m'), & \text{if } m \in M_1 - M_2, \\ S_{\underset{\sim}{L_2}}(m) - \bigcup_{m'<m} S_{\underset{\sim}{L_1}}(m'), & \text{if } m \in M_2 - M_1, \\ ((S_{\underset{\sim}{L_1}}(m) \bigcup S_{\underset{\sim}{L_2}}(m)) - \bigcup_{m'<m} S_{\underset{\sim}{L_1}}(m')) - \bigcup_{m''<m} S_{\underset{\sim}{L_2}}(m''), & \text{if } m \in M_1 \cap M_2; \end{cases}$$

(3) Complement: ($M = \{1 - m \mid m \in M_1\}$)

$$S_{\underset{\sim}{L}}(m) = S_{\underset{\sim}{L_1}}(1 - m),$$

(4) Concatenation:

$$S_{\underset{\sim}{L}}(m) = \bigcup_{\substack{min(m_1, m_2) = m \\ m_1 \in M_1 \\ m_2 \in M_2}} S_{\underset{\sim}{L_1}}(m_1) S_{\underset{\sim}{L_2}}(m_2) - \bigcup_{\substack{min(m_1', m_2') > m \\ m_1 \in M_1 \\ m_2 \in M_2}} S_{\underset{\sim}{L_1}}(m_1') S_{\underset{\sim}{L_2}}(m_2').$$

(5) Star: assuming that $M_1 = \{m_1, \ldots, m_n\}$ and $1 \geq m_1 > m_2 > \ldots > m_n \geq 0$,

$$S_{\underset{\sim}{L}}(m_1) = (S_{\underset{\sim}{L_1}}(m_1))^* \text{ if } m_1 = 1,$$

$$S_{\underset{\sim}{L}}(1) = \{\lambda\} \text{ and } S_{\underset{\sim}{L}}(m_1) = (S_{\underset{\sim}{L_1}}(m_1))^+ - \{\lambda\} \text{ if } m_1 \neq 1,$$

$$S_{\underset{\sim}{L}}(m_i) = (\bigcup_{j \leq i} S_{\underset{\sim}{L_1}}(m_j))^+ - \bigcup_{k<i} S_{\underset{\sim}{L}}(m_k) - \{\lambda\}, \quad 1 < i \leq n.$$

It is clear that all the sets defined above are regular. So, we have finished this proof. □

3 Finite fuzzy automata

The reader may refer to [8, 3] for basic definitions in automata theory.

Definition 3.1 *A nondeterministic finite automaton with fuzzy transitions (FT-NFA) $\widetilde{A}$ is a 5-tuple $\widetilde{A} = (Q, \Sigma, \tilde{\delta}, s, F)$, where*
Q is the finite set of states;
Σ is the finite set of input symbols;

$\tilde{\delta}: Q \times \Sigma \times Q \to [0,1]$ is the degree function of state transitions;

s is the initial state; and

$F \subseteq Q$ is the set of final states.

For $x \in \Sigma^*$ and $p, q \in Q$, define

$$\tilde{\delta}^*(p,x,q) = \begin{cases} 0, & \text{if } x = \lambda \text{ and } p \neq q, \\ 1, & \text{if } x = \lambda \text{ and } p = q, \\ \max_{r \in Q}\{min\{\tilde{\delta}^*(p,x',r),\ \tilde{\delta}(r,a,q)\} \mid x = x'a,\ x' \in \Sigma^*,\ a \in \Sigma\}, & \text{otherwise.} \end{cases}$$

Then we say that $x \in \Sigma^*$ is accepted by $\widetilde{A}$ with degree $d_{\widetilde{A}}(x)$, where

$$d_{\widetilde{A}}(x) = max\{\tilde{\delta}^*(s,x,q) \mid q \in F\}.$$

We denote by $\widetilde{L}\,(\widetilde{A})$ the set:

$$\widetilde{L}\,(\widetilde{A}) = \{(x, d_{\widetilde{A}}(x)) \mid x \in \Sigma^*\}.$$

Note that by the above definition, the value of $d_{\widetilde{A}}(\lambda)$ only can be either 1 or 0 for any FT-NFA $\widetilde{A}$. In many discussions, this restriction is easily understood but can be cumbersome to describe. So, if there is no special mentioning when considering problems related to FT-NFA in the sequel, the reader should assume that λ and its degree are not considered.

Example 3.1 *Let $\Sigma = \{a, b\}$ and an FT-NFA $\widetilde{A}_1$ be defined in the following:*

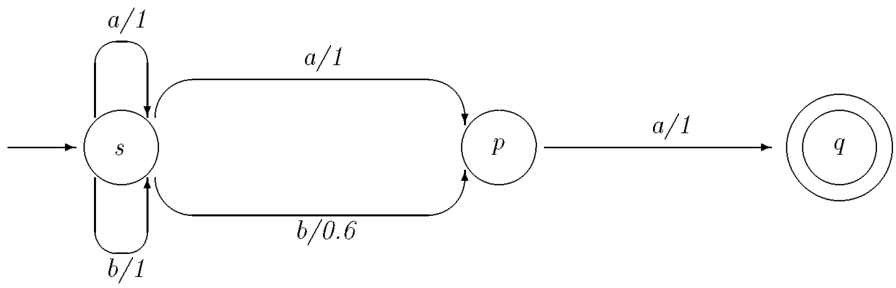

Figure 1

Obviously,

$$\tilde{L}(\tilde{A}_1) = \{(x,1) \mid x \in \{a,b\}^*aa\} \cup \{(y,0.6) \mid y \in \{a,b\}^*ba\}.$$

We omit the pairs whose second components are 0.

FT-NFA are a special type of Mealy machines, where the output has a special meaning. They are also a special class of the maximin automata introduced by Santos in [9]. In an FT-NFA, the set of final states is a crisp set (which, of course, is a special case of fuzzy sets) and the initial state is crisp, too.

Definition 3.2 *A* <u>*deterministic*</u> *finite automaton with fuzzy transitions (FT-DFA)* $\tilde{A} = (Q, \Sigma, \tilde{\delta}, s, F)$ *is an FT-NFA with the condition that for each* $p \in Q$ *and* $a \in \Sigma$, *if* $\tilde{\delta}(p,a,q) > 0$ *and* $\tilde{\delta}(p,a,q') > 0$, *then* $q = q'$.

Theorem 3.1 $\tilde{L}$ *is a regular fuzzy language iff* $\tilde{L}$ *is accepted by an FT-NFA* $\tilde{A}$ *with the exception of* λ.

Proof. Let $\tilde{L}$ be a regular fuzzy language and $f_{\tilde{L}} : \Sigma^* \to M$ be the membership function. Then $M = \{m_1, \ldots, m_n\}$ for some $n \geq 1$ and $S_{\tilde{L}}(m_i)$ is regular for each $m_i \in M$. Note that $S_{\tilde{L}}(m_i) \cap S_{\tilde{L}}(m_j) = \emptyset$ for $i \neq j$ since $f_{\tilde{L}}$ is a function.

Let $A_i = (Q_i, \Sigma, \delta_i, s_i, F_i)$ be a DFA (or an NFA) such that $S_{\tilde{L}}(m_i) = L(A_i)$, $1 \leq i \leq n$. We construct $\tilde{A}_i = (Q_i, \Sigma, \tilde{\delta}_i, s_i, F_i)$ where

$$\tilde{\delta}_i(p,a,q) = \begin{cases} m_i, & \text{if } (p,a,q) \in \delta_i, \\ 0, & \text{otherwise.} \end{cases}$$

We assume that $Q_i \cap Q_j = \emptyset$ for $i \neq j$. Define $\tilde{A} = (Q, \Sigma, \tilde{\delta}, s, F)$ such that

$$Q = Q_1 \cup \ldots \cup Q_n \cup \{s\} \quad \text{and} \quad s \notin Q_1 \cup \ldots \cup Q_n,$$

$$F = F_1 \cup \ldots \cup F_n,$$

$$\tilde{\delta}(p,a,q) = \begin{cases} \tilde{\delta}_i(p,a,q) & \text{if } p, q \in Q_i \text{ for some } i \in \{1,\ldots,n\}, \\ \tilde{\delta}_i(s_i,a,q) & \text{if} p = s \text{ and } q \in Q_i \text{ for some } i \in \{1,\ldots,n\}, \\ 0 & \text{otherwise.} \end{cases}$$

Clearly, $\tilde{A}$ accepts $\tilde{L}$ with the possible exception of λ .

Let $\tilde{A} = (Q, \Sigma, \tilde{\delta}, s, F)$ be an FT-NFA. Define a fuzzy language $\tilde{L}$ with $f_{\tilde{L}}(w) = d_{\tilde{A}}(w)$ for each $w \in \Sigma^*$, $(f_{\tilde{L}}(\lambda) = 0)$. We now show that $\tilde{L}$ is a regular fuzzy language.

Let $M = \{m \mid \tilde{\delta}(p,a,q) = m \text{ for some } p,q \in Q, a \in \Sigma\}$. Obviously, M is finite. Assume that $M = \{m_1, \ldots, m_n\}$ with $m_1 > m_2 > \ldots > m_n, n \geq 1$. For each i, $1 \leq i \leq n$, define an NFA

$$A_i = (Q, \Sigma, \delta_i, s, F)$$

where $\delta_i = \{(p,a,q) \mid \tilde{\delta}(p,a,q) \geq m_i\}$. Define the languages L_i, $1 \leq i \leq n$, in the increasing sequence of i as follows:

$$L_1 = L(A_1),$$

$$L_i = L(A_i) - \bigcup_{j=1}^{i-1} L_j.$$

Then $S_{\tilde{L}}(m_i) = L_i$ and L_i is a regular language, for each i, $1 \leq i \leq n$. Therefore, $\tilde{L}$ is a regular fuzzy language. □

Theorem 3.2 *Let $\tilde{L}$ be a regular fuzzy language. Then $\tilde{L}$ is accepted by an FT-DFA iff it satisfies the following condition: For $x, y \in \Sigma^+, u \in \Sigma^*$*

$$(*) \qquad x = yu \text{ and } f_{\tilde{L}}(y) > 0 \text{ implies that } f_{\tilde{L}}(x) \leq f_{\tilde{L}}(y).$$

Proof. Let $\tilde{L}$ be accepted by an FT-DFA $\tilde{A}= (Q, \Sigma, \tilde{\delta}, s, F)$. We show that $\tilde{L}$ satisfies $(*)$. Let $x = yu$, $x, y \in \Sigma^+$ and $a \in \Sigma^*$. If $d_{\tilde{A}}(x) = 0$, then $f_{\tilde{L}}(x) \leq f_{\tilde{L}}(y)$ is true trivially. Otherwise,

$$f_{\tilde{L}}(x) = d_{\tilde{A}}(x) = min\{\tilde{\delta}^*(s,y,q),\ \tilde{\delta}^*(q,u,f)\} \leq \tilde{\delta}^*(s,y,q) = d_{\tilde{A}}(y) = f_{\tilde{L}}(y)$$

where $q, f \in F$.

For the other direction of the proof, let $\tilde{L}$, with $f_{\tilde{L}} : \Sigma^* \to M$, satisfy the condition $(*)$. Assume that $M = \{m_1, \ldots, m_n\}$. It is clear that we can construct a DFA $A_i = (Q_i, \Sigma, \delta_i, s_i, F_i)$ such that $L(A_i) = S_{\tilde{L}}(m_i)$ for each i, $1 \leq i \leq n$. Note that for $1 \leq i, j \leq n$ and $i \neq j$,

$$L(A_i) \cap L(A_j) = S_{\tilde{L}}(m_i) \cap S_{\tilde{L}}(m_j) = \emptyset$$

Now we construct a DFA $A = (Q, \Sigma, \delta, s, F)$ where $Q = Q_1 \times \ldots \times Q_n$, $s = (s_1, \ldots, s_n)$, $\delta : Q \times \Sigma \to Q$ is defined by $\delta((q_1, \ldots, q_n), a) = (\delta_1(q_1, a), \ldots, \delta_n(q_n, a))$, and $F = F_1' \cup \cdots \cup F_n'$ where $F_i' = \{(q_1, \ldots, q_n) \in Q \mid q_i \in F_i \text{ and } q_j \notin F_j \text{ for } j \neq i\}$, $1 \leq i \leq n$. It is clear that $F_i' \cap F_j' = \emptyset$, for $i \neq j$, and

$$S_{\tilde{L}}(m_i) = \{w \in \Sigma^* \mid \delta^*(s, w) \in F_i'\}.$$

Based on the above DFA A, we define an FT-DFA $\underset{\sim}{A}= (Q, \Sigma, \tilde{\delta}, s, F)$ such that

$$\tilde{\delta}(p, a, q) = \begin{cases} m_i & \text{if } \delta(p, a) = q \in F'_i, \\ 1 & \text{if } \delta(p, a) = q \notin F, \\ 0 & \text{otherwise.} \end{cases}$$

It remains to show that $d_{\underset{\sim}{A}}(w) = f_{\underset{\sim}{L}}(w)$, for each $w \in \Sigma^+$. But first we show that $\underset{\sim}{A}$ has the following property: For each $w \in \Sigma^+$ with $w = xa$, $x \in \Sigma^*$ and $a \in \Sigma$,

$(**)$ $\quad d_{\underset{\sim}{A}}(w) = m_i > 0$ **iff** $\delta^*(s, x, p) \geq m_i$ and $\delta(p, a, q) = m_i$ for some $q \in F_i, 1 \leq i \leq n$.

The *if* part holds obviously. For the *only if* part, it holds trivially when $x = \lambda$. For $x \neq \lambda$, we assume the contrary, i.e., $\delta^*(s, x, p) = m_i$ and $\delta(p, a, q) = m_j > m_i$. Then there exists a decomposition of $x = ybz$, $y, z \in \Sigma^*$ and $b \in \Sigma$, such that $\delta^*(s, y, r) \geq m_i$, $\delta(r, b, t) = m_i$, and $\delta(t, z, p) \geq m_i$. By the definition of $\underset{\sim}{A}$, we know that $t \in F_i$ and $q \in F_j$. Thus, we have $f_{\underset{\sim}{L}}(yb) = m_i$ and $f_{\underset{\sim}{L}}(w) = m_j$. Since we assume that $m_j > m_i$, this is a contradiction to $(*)$. So, $(**)$ holds. Furthermore, the righthand side of $(**)$ implies that $xa \in S_{\underset{\sim}{L}}(m_i)$, i.e., $f_{\underset{\sim}{L}}(w) = m_i$. Therefore, we have finished the proof. $\Box$

Indeed, the condition (*) can be interpreted as follows. We have n regular languages $R_1, \ldots, R_n$, associated with the decreasing sequence $m_1 > m_2 > \ldots > m_n$. Whenever a word $y \in R_j$ is a prefix of a word $x \in R_i$, then $i \geq j$. This makes the construction possible. The construction does not work, for instance, for the two regular languages $\{a, aba\}$ and $\{ab\}$.

The condition $(*)$ can be used to show that some regular fuzzy languages are not accepted by any FT-DFA.

Corollary 3.1 *The family of fuzzy languages accepted by FT-DFA is properly included in the family of fuzzy languages accepted by FT-NFA.*

Proof. The inclusion is obvious. We only need to show that the inclusion is proper. Consider

$$\underset{\sim}{L}= \{a/0.5, ab/1\}.$$

Obviously, $\underset{\sim}{L}$ is accepted by an FT-NFA but not by any FT-DFA by Theorem 3.2.

Finite automata with fuzzy transitions are apparently a natural model for regular fuzzy languages. Many similar models have been studied in the past. However, the difference in accepting power between its deterministic and nondeterministic version and the fact that many very simple regular fuzzy languages are not accepted by its deterministic version make it unfavorable to be used in practice. An FT-NFA can be represented in a matrix form. However, it may be feasible only for FT-NFA with a small number of states.

Naturally, another finite-fuzzy-automaton model is a special class of Mooore machines. This model also characterizes the family of regular fuzzy languages. But unlike the fuzzy transition model, its deterministic and nondeterministic versions are equivalent in accepting power. This model is also simpler and easier to implement than the previous model.

Definition 3.3 *A* <u>*nondeterministic*</u> *finite automaton with fuzzy (final) states (FS-NFA or FS-FA)* $\widetilde{A}$ *is a 5-tuple* $\widetilde{A} = (Q, \Sigma, \delta, s, \widetilde{F}_{\tilde{A}})$ *where* Q, Σ, δ, *and* s *are the same as in an NFA, and* $\widetilde{F}_{\tilde{A}} : Q \to [0,1]$ *is the degree function for the fuzzy final-state set.*

Define

$$d_{\underset{A}{\sim}}(x) = max\{\widetilde{F}_{\tilde{A}}(q) \mid (s, x, q) \in \delta^*\}.$$

Note that δ^* *is the transitive and reflexive closure of* δ *defined as for a normal NFA. Then we say that* x *is accepted by* $\widetilde{A}$ *with degree* $d_{\underset{A}{\sim}}(x)$*. The fuzzy language accepted by* $\widetilde{A}$*, denoted* $\widetilde{L}(\widetilde{A})$*, is the set* $\{(x, d_{\underset{A}{\sim}}(x)) \mid x \in \Sigma^*\}$.

Example 3.2 *Let* $\Sigma = \{a, b\}$*. An FS-NFA* $\widetilde{A}$ *is the following:*

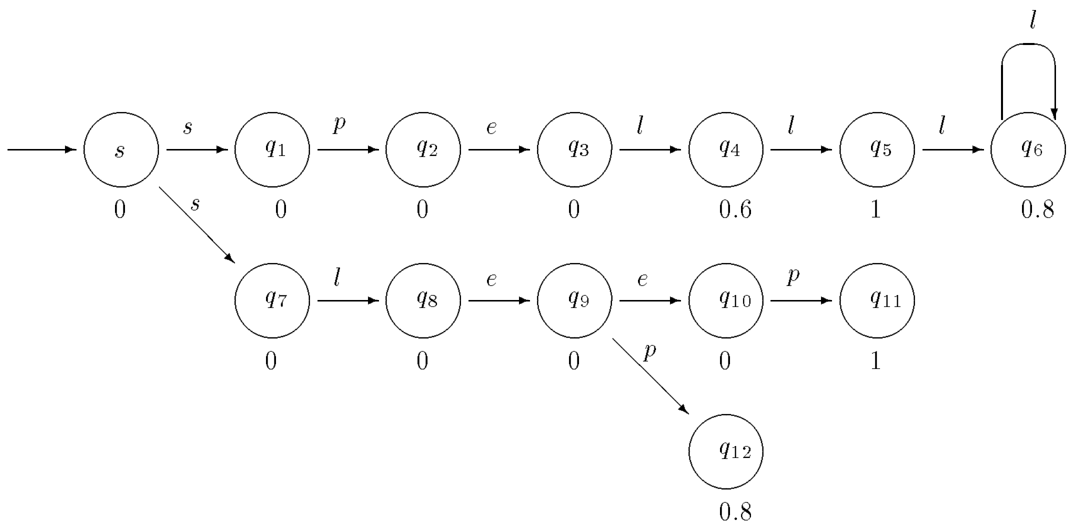

Figure 2

Then $d_{\underset{\sim}{A}}(sleep) = 1$, $d_{\underset{\sim}{A}}(spellllll) = 0.8$, *and* $d_{\underset{\sim}{A}}(sle) = 0$.

Definition 3.4 *A* <u>*deterministic*</u> *finite automaton with fuzzy states (FS-DFA)* $\widetilde{A} = (Q, \Sigma, \delta, s, \widetilde{F}_{\underset{\sim}{A}})$ *is an FS-NFA with* δ *being a function* $Q \times \Sigma \to Q$ *instead of a relation. Hence, for each* $x \in \Sigma^*$, $d_{\underset{\sim}{A}}(x) = \widetilde{F}_{\underset{\sim}{A}}(q)$ *where* $q = \delta^*(s, x)$.

Define $d_{\underset{\sim}{A}}(x) = 0$ *if* $\delta^*(s, x)$ *is not defined.*

Theorem 3.3 *Let* $\widetilde{L}$ *be a fuzzy language. Then* $\widetilde{L}$ *is a regular fuzzy language iff it is accepted by an FS-DFA.*

Proof. Let $f_{\underset{\sim}{L}} : \Sigma^* \to M$ be the membership function of $\widetilde{L}$. Assume that $\widetilde{L}$ is a regular fuzzy language. Then M is finite and, for each $m \in M$, $S_{\underset{\sim}{L}}(m)$ is a regular set. Assume that $M = \{m_1, \ldots, m_n\}$. We construct a DFA $A_i = (Q_i, \Sigma, \delta_i, s_i, F_i)$ for each i, $1 \leq i \leq n$, such that $L(A_i) = S_{\underset{\sim}{L}}(m_i)$. Define an FS-DFA $\widetilde{A}= (Q, \Sigma, \delta, s, \widetilde{F}_{\underset{\sim}{A}})$ to be the cross product of $A_1, \ldots, A_n$ with

$$\widetilde{F}_{\underset{\sim}{A}}((q^{(1)}, \ldots, q^{(n)})) = \begin{cases} m_i, & q^{(i)} \in F_i \text{ for some } i,\ 1 \leq i \leq n, \text{ and } q^{(j)} \notin F_j, \text{ for all } j \neq i \\ 0, & \text{otherwise.} \end{cases}$$

Note that if $(q^{(1)}, \ldots, q^{(n)})$ is reachable from $(s_1, \ldots, s_n)$ in $\widetilde{A}$, then it is impossible to have $q^{(i)} \in F_i$ and $q^{(j)} \in F_j$ for $i \neq j$ since $L(A_i) \cap L(A_j) = \emptyset$ for $i \neq j$, $1 \leq i, j \leq n$. Obviously, $\widetilde{A}$ accepts $\widetilde{L}$.

For the other direction of the proof, let $\widetilde{A}= (Q, \Sigma, \delta, s, \widetilde{F}_{\underset{\sim}{A}})$ be an FS-DFA. Define

$$M = \{m \mid \widetilde{F}_{\underset{\sim}{A}}(q) = m \text{ for some } q \in Q\}.$$

M is a finite set. For each $m \in M$, define

$$A_m = (Q, \Sigma, \delta, s, F_m)$$

where $F_m = \{q \mid \widetilde{F}_{\underset{\sim}{A}}(q) = m\}$. Let $\widetilde{L} = \widetilde{L}(\widetilde{A})$, i.e. $f_{\underset{\sim}{L}} = d_{\underset{\sim}{A}}$. Then clearly, for each $m \in M$, $S_{\underset{\sim}{L}}(m) = L(A_m)$ is regular. $\widetilde{L}$ is a regular fuzzy language. □

Theorem 3.4 *A fuzzy language is accepted by an FS-NFA iff it is accepted by an FS-DFA.*

Proof. It suffices to show that if $\widetilde{L}=\widetilde{L}(\widetilde{A})$ for an FS-NFA $\widetilde{A}$ then $\widetilde{L}=\widetilde{L}(\widetilde{A}')$ for some FS-DFA $\widetilde{A}'$. Let $\widetilde{A}= (Q, \Sigma, \delta, s, \widetilde{F}_{\underset{\sim}{A}})$. The construction of $\widetilde{A}' =$

$(Q', \Sigma, \delta', s', \widetilde{F}'_{\widetilde{A}})$ is straightforward. We can just use the standard subset-construction method and, for each $P \in Q'$ $(P \subseteq Q)$, define

$$\widetilde{F}'_{\widetilde{A}}(P) = max\{m \mid m = \widetilde{F}_{\widetilde{A}}(q), q \in P\}.$$

It is clear that $\widetilde{L} = \widetilde{L}(\widetilde{A}')$. $\Box$

Example 3.3 *Let an FS-NFA be defined by:*

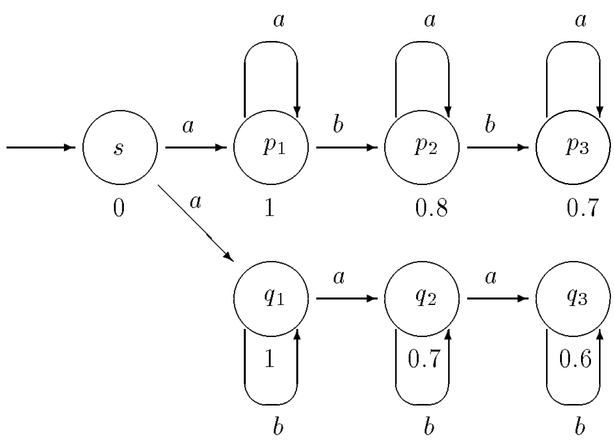

Figure 3

Define $r_1 = \{p_1, q_1\}$, $r_2 = \{p_1, q_2\}$, $r_3 = \{p_2, q_1\}$, $r_4 = \{p_1, q_3\}$, $r_5 = \{p_2, q_2\}$, $r_6 = \{p_3, q_1\}$, $r_7 = \{p_2, q_3\}$, $r_8 = \{p_3, q_2\}$, *and* $r_9 = \{p_3, q_3\}$. *Then an FS-DFA that is equivalent to the above FS-NFA is given in the following:*

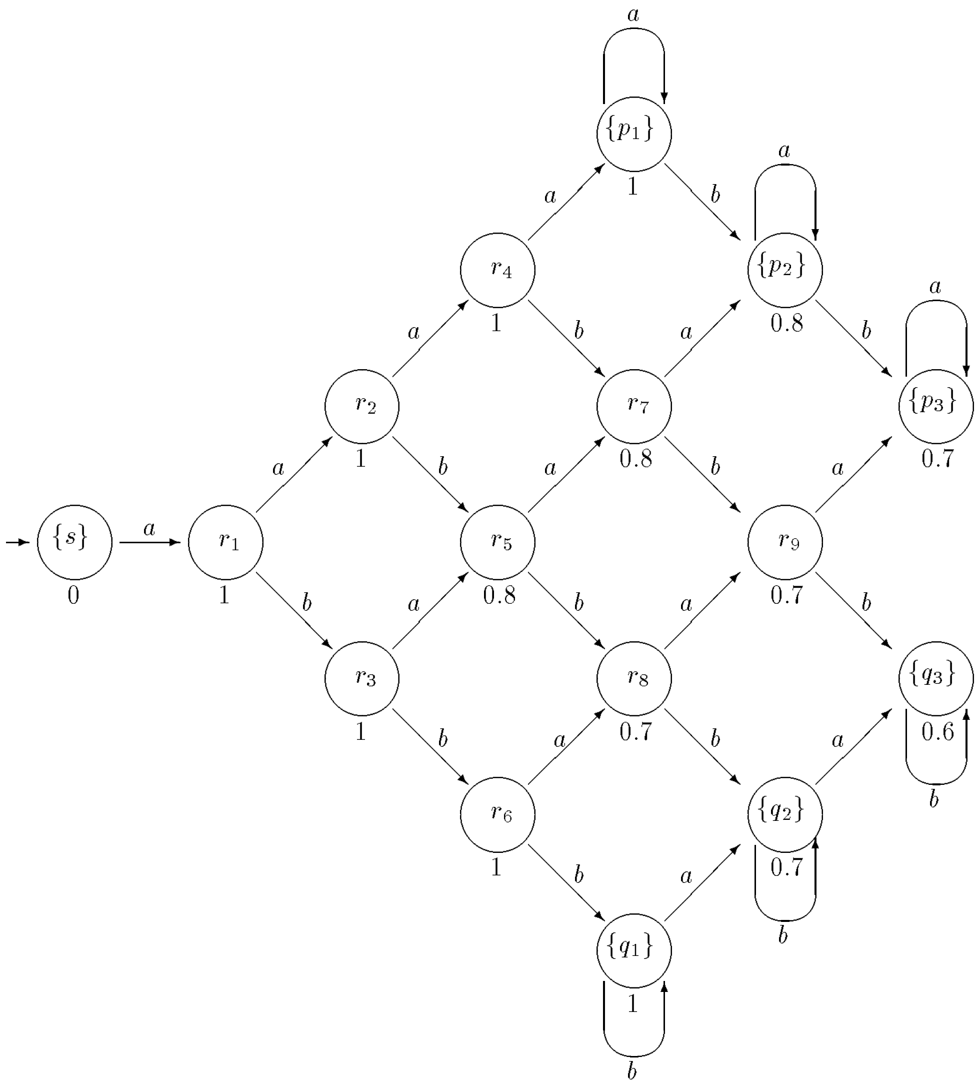
{s}
0
r1
1
r2
1
r3
1
r4
1
r5
0.8
r6
1
r7
0.8
r8
0.7
r9
0.7
{p1}
1
{p2}
0.8
{p3}
0.7
{q1}
1
{q2}
0.7
{q3}
0.6
a
b

Figure 4

An extension of the Myhill-Nerode Theorem is given below, which can be easily proved.

Theorem 3.5 *(The extended Myhill-Nerode theorem) The following three statements are equivalent:*

(i) $\tilde{L}$ *is a regular fuzzy language over* Σ *.*

(ii) $\tilde{L}$ *is the union of some of the equivalence classes of a right invariant equivalence relation of finite index.*

(iii) Let the relation $R_{\tilde{L}} \subseteq \Sigma^* \times \Sigma^*$ *be defined by* $x R_{\tilde{L}} y$ *iff for all* $z \in \Sigma^*$, $f_{\tilde{L}}(xz) = f_{\tilde{L}}(yz)$. *Then* $R_{\tilde{L}}$ *is an equivalence relation of finite index.*

The minimization algorithm for DFA described in [3] can also be extended for FS-DFA as follows:

ALGORITHM 3.1

Let $\tilde{A} = (Q, \Sigma, \delta, q_0, \tilde{F}_{\tilde{A}})$ be an FS-DFA. Assume that $Q = \{q_0, \ldots, q_n\}$, $n \geq 0$, and let $P = \{(q_i, q_j) \mid q_i, q_j \in Q$ and $0 \leq i < j \leq n\}$.

begin

1) **for** each pair $(q_i, q_j) \in P$, and $\tilde{F}_{\tilde{A}}\ (q_i) \neq \tilde{F}_{\tilde{A}}\ (q_j)$ **do** mark (q_i, q_j);

2) **for** each unmarked pair $(p, q) \in P$ **do**

if for some $a \in \Sigma$, $(\delta(p, a), \delta(q, a))$ is marked then

begin

mark (p, q);

recursively mark all unmarked pairs on the list of (p, q) and

on the lists of other pairs that are marked at this step.

end

else

for all input symbols $a \in \Sigma$ **do**

put (p, q) on the list for $(\delta(p, a), \delta(q, a))$ unless $\delta(p, a) = \delta(q, a)$

end

We omit the proof that the FS-DFA constructed with the above algorithm is minimal in terms of the number of states.

4 Fuzzy regular expressions (FRE)

For each regular fuzzy language, there is a finite number of degrees and the set of all words that are associated with each degree is a regular language. Therefore, we can naturally represent a regular fuzzy language by a modified regular expression like the following:

$$ab^*/0.6 + ab^*a/1 + (bab^* + bbab^*)/0.8$$

The semantics of the expression is clear. We give a formal definition for fuzzy regular expressions (FREs) in the following:

Definition 4.1 *Let Σ be a finite alphabet and M a finite set of real numbers in $[0, 1]$.*

1) Let e be a regular expression over Σ and $m \in M$. Then $(e)/m$ is a fuzzy regular expression.

2) Let $\widetilde{e}_1$ and $\widetilde{e}_2$ be fuzzy regular expressions. Then $\widetilde{e}_1 + \widetilde{e}_2$, $(\widetilde{e}_1) \cdot (\widetilde{e}_2)$, and $(\widetilde{e}_1)^*$ are all fuzzy regular expressions.

3) A fuzzy regular expression is formed by applying 1) and 2) a finite number of times.

Definition 4.2 *A fuzzy regular expression over Σ is normalized if it is of the following form*

$$e_1/m_1 + e_2/m_2 + \ldots + e_n/m_n$$

where $e_1, e_2, \ldots, e_n$ are regular expressions over Σ and $m_1, m_2, \ldots, m_n$ are numbers in $[0, 1]$, $n \geq 1$.

Note that if $m = 1$ then e/m can simply be written as e. We assume that "$\cdot$" and "$*$" have higher priorities than "/". So, certain pairs of parentheses can be omitted.

Example 4.1 *The following are all valid FRE's:*

(1) $a^*/1 + a^*ba^*/0.8 + a^*ba^*ba^*/0.5$,

(2) $(b^*ab^*/0.7) \cdot (a^*ba^*/0.5) + b^*$,

(3) $abba + baab + (a + b)^*a(a + b)^*b(a + b)^*$,

where both (1) and (3) are normalized. The following are not valid FRE's:

(4) $(a^*/0.5)/0.7 + a^*ba^*/1$,

(5) $(ba^*b/0.2)(a^*)/0.5 + (ab + a)^*/1$,

(6) $(ab/0.9 + b/0.5)^*/0.9 + (aba)^*/1$.

Definition 4.3 *An FRE $\widetilde{e}$ is called a strictly normalized FRE if it is normalized, i.e.,*

$$\widetilde{e} = e_1/m_1 + e_2/m_2 + \ldots + e_n/m_n,$$

and for any $m_i \neq m_j$, $L(e_i) \cap L(e_j) = \emptyset$.

Definition 4.4 *Let $\widetilde{e}$ be an FRE. Then $\widetilde{L}(\widetilde{e})$ is defined by:*

(1) *if $\widetilde{e} = e/m$ where e is a regular expression, then $\widetilde{L}(\widetilde{e}) = \{(x, m) \mid x \in L(e)\}$;*

(2) *if $\widetilde{e} = \widetilde{e}_1 + \widetilde{e}_2$, $\widetilde{e} = (\widetilde{e}_1) \cdot (\widetilde{e}_2)$, or $\widetilde{e} = (\widetilde{e}_1)^*$, then $\widetilde{L}(\widetilde{e}) = \widetilde{L}(\widetilde{e}_1) \cup \widetilde{L}(\widetilde{e}_2)$, $\widetilde{L}(\widetilde{e}) = \widetilde{L}(\widetilde{e}_1) \cdot \widetilde{L}(\widetilde{e}_2)$, or $\widetilde{L}(\widetilde{e}) = (\widetilde{L}(\widetilde{e}_1))^*$, respectively.*

It is easy to show that the families of languages represented by FREs, normalized FREs, and strictly normalized FRE's, respectively, are equivalent, and they all coincide with the family of fuzzy regular languages.

Example 4.2 *The following FREs are equivalent:*

(1) $(b^*/1)(b/0.5) + ((a+b)^*a(a+b)^*/0.5)(b/1) + ((a+b)^*aa(a+b)^*/0.8)(b/1)$,

(2) $b^*b/0.5 + (a + b)^*a(a + b)^*b/0.5 + (a + b)^*aa(a + b)^*b/0.8$,

(3) $(a + \lambda)(b + ba)^*b/0.5 + (a + b)^*aa(a + b)^*b/0.8$,

The reader can verify that (2) and (3) are normalized and (3) is strictly normalized.

5 Marked regular fuzzy languages

In lexical analysis, it is a common practice to construct one automaton for accepting several or many different tokens (i.e., regular languages). In order to distinguish strings belonging to different tokens, final states are marked with token names. Often, there is a linear order of priorities associated with the token names. Strings belonging to two or more tokens are marked with the name that has the highest priority among them. This idea appears to be especially useful when it is applied to fuzzy languages. We formulate this with the following definitions.

Definition 5.1 *Let $\widetilde{L}$ be a fuzzy language, T a finite set of names with a linear order $<$, and $\mu : \Sigma^* \to T$ a function. Then we call the set*

$$\{(w, f_{\widetilde{L}}(w), \mu(w)) \mid w \in \Sigma^*\}$$

a marked fuzzy language, denoted $(\widetilde{L}, \mu)$, and μ the marking function of the language.

Definition 5.2 *Let $(\widetilde{L}, \mu)$ with $\mu : \Sigma^* \to T$ be a marked fuzzy language. Then $(\widetilde{L}, \mu)$ is called a marked regular fuzzy language if the following two conditions hold:*

1) $\widetilde{L}$ is a regular fuzzy language.

2) For each $t \in T$, $\mu^{-1}(t) = \{w \in \Sigma^* \mid \mu(w) = t\}$ is a regular language.

Note that if $f_{\widetilde{L}}(w) = 0$ for some $w \in \Sigma^*$, then the value of $\mu(w)$ is unimportant.

Definition 5.3 *Let $(\widetilde{L}_1, \mu_1)$ and $(\widetilde{L}_2, \mu_2)$ be two marked fuzzy languages over an alphabet Σ. We say that $(\widetilde{L}_1, \mu_1)$ and $(\widetilde{L}_2, \mu_2)$ are equivalent, denoted $(\widetilde{L}_1, \mu_1) = (\widetilde{L}_2, \mu_2)$, if*

1) $\widetilde{L}_1 = \widetilde{L}_2$, and

2) $\mu_1(w) = \mu_2(w)$ for all $w \in \Sigma^*$ such that $f_{\widetilde{L}_1}(w) = f_{\widetilde{L}_2}(w) \neq 0$.

Similarly, we say that $(\widetilde{L}_1, \mu_1)$ is included in $(\widetilde{L}_2, \mu_2)$ if

1) $\widetilde{L}_1 \subseteq \widetilde{L}_2$, and

2) $\mu_1(w) = \mu_2(w)$ for all $w \in \Sigma^*$ such that $f_{\widetilde{L}_1}(w) \neq 0$.

Marked union, i.e. union of marked fuzzy languages, is a useful tool in lexical analysis. For example, a string x belongs to token t_1 with 0.9 degree and also to token t_2 with 0.6 degree. Then in the marked union of the two tokens, x is considered to be marked with t_1. If the two degrees are equal, then x is marked with the token that has a higher priority. Formally, we give the following definition.

Definition 5.4 (*Marked union*) *Let $(\widetilde{L}_1, \mu_1)$ and $(\widetilde{L}_2, \mu_2)$ be two marked fuzzy languages where $\mu_1, \mu_2 : \Sigma^* \to T$. A marked fuzzy language $(\widetilde{L}, \mu)$ is called the marked union of $(\widetilde{L}_1, \mu_1)$ and $(\widetilde{L}_2, \mu_2)$ if $\widetilde{L} = \widetilde{L}_1 \cup \widetilde{L}_2$ and $\mu : \Sigma^* \to T$ is defined by*

$$\mu(w) = \begin{cases} \mu_1(w), & \text{if } f_{\widetilde{L}_1}(w) > f_{\widetilde{L}_2}(w) \text{ } or \text{ } f_{\widetilde{L}_1}(w) = f_{\widetilde{L}_2}(w) \text{ and } \mu_2(w) < \mu_1(w), \\ \mu_2(w), & \text{otherwise} \end{cases}$$

for each $w \in \Sigma^*$.

Example 5.1 *Let* $T = \{ID, INT\}$ *with* $INT < ID$, $\Sigma_l = \{a, \ldots, z\}$, $\Sigma_d = \{0, \ldots, 9\}$, *and* $\Sigma = \Sigma_l \cup \Sigma_d$. $\widetilde{L}_1$ *is defined, informally, by*

$$\Sigma_l \Sigma^*/1 + \Sigma_d \Sigma^* \Sigma_l \Sigma^* \Sigma_l \Sigma^*/0.9 + \Sigma_d \Sigma^* \Sigma_l \Sigma^*/0.5$$

and $\widetilde{L}_2$ *by*

$$\Sigma_d \Sigma_d^*/1 + \Sigma_d \Sigma_d^* \Sigma_l \Sigma_d^*/0.7.$$

Let $\mu_1, \mu_2 : \Sigma^* \to T$ *be defined by*

$$\mu_1(x) = ID \text{ and } \mu_2(x) = INT$$

for all $x \in \Sigma^*$. *Let* $(\widetilde{L}, \mu)$ *be the marked union of* $(\widetilde{L}_1, \mu_1)$ *and* $(\widetilde{L}_2, \mu_2)$. *Then* $\mu(32h01) = INT$ *and* $\mu(132n4p) = ID$.

6 A fuzzy lexical analyser

The lexical analyser LEX is available in almost all versions of UNIX. Here we propose a "fuzzy" extension of LEX, named FLEX, in the following.

FLEX is a fuzzy lexical analyser generator. All features of LEX work in FLEX except that the symbol "/" in the expressions should be written as "\/" now.

FLEX has the following two additional features:

(1) Any LEX expression can be followed by a "/" and a number between 0 and 1. For example,

$$[0-9a-zA-Z]+/0.75+[.\,;?!]/1.$$

The "$/n$" part, where n is a number between 0 and 1, is called the degree of the expression. Degrees cannot be nested, i.e., if an expression is specified with a degree, then none of its subexpressions is allowed to be specified with a degree. For example, $\{a/0.6\}bc/0.7$ is an invalid expression. Degrees cannot be specified within a pair of "[" and "]".

(2) Besides the three parts in a LEX program, a fourth part can be used to define actions for different ranges of degrees. For example,

$[0.8, 1]:$ ACCEPT ;

$[0.6, 0.8):$ QUESTION ;

$[0, 0.6):$ REJECT ;

where ACCEPT, QUESTION, and REJECT are all key words for FLEX denoting, respectively, that

(a) the string is accepted as the token;

(b) an on-line question is given to the user; the string is accepted if the user answers "yes", rejected if the user answers "no";

(c) the string is rejected,

If the fourth part is not given, the following rules are assumed by default:

$[1,1]:$ ACCEPT;

$[0.9,1):$ {WARNING; ACCEPT};

$[0.5,0.9):$ QUESTION;

$[0,0.5):$ REJECT;

Example 6.1 *An FLEX file for a student-mark handling program is the following:*

```
%%
%{
#include  "type.h"
#include  "yy.tab.h"
%}

%%
list+lis/0.9+(lst+ls)/0.8            { ...... }
enter+(ente+ent)/0.9+(nter+entr)/0.8 { ...... }
print+(prt+prnt)/0.9+rint/0.6        { ...... }
calc+(cal+comp)/0.9+(ca+com)/0.6     { ...... }
./0.0
%%

[0.9 , 1] :   ACCEPT;
[0.8 , 0.9) : {WARNING; ACCEPT};
[0.6 , 0.8) : QUESTION;
[0 , 0.6) :   REJECT;
```

References

[1] D. Dubois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, 1980, San Diego.

[2] N. Honda, M. Nasu and S. Hirose, "F-Recognition of Fuzzy Languages", *Fuzzy Automata and Decision Processes, edited by M.M. Gupta, G.N. Saridis and B.R. Gaines*, North-Holland, 1977, 149-168.

[3] J. Hopcroft and J. Ullman, *Introduction to Automata Theory, Languages, and Computation,* Addison-Wesley, 1979.

[4] E.T. Lee and L.A. Zadeh, "Note on Fuzzy Languages ", *Information Sciences*, 1 (1969) 421-434.

[5] M. Mizumoto, J. Toyoda, and K. Tanaka, "Fuzzy Languages", *Systems, Computers, Controls* 1 (1970) 36-43.

[6] M. Mizumoto, J. Toyoda, and K. Tanaka, "Various Kinds of Automata with Weights", *Journal of Computer and System Sciences* 10 (1975) 219-236.

[7] M. Nasu and N. Honda, "Fuzzy events realized by finite probabilistic automata", *Information and Control* 12 (1968) 284-303.

[8] A. Salomaa, *Theory of Automata*, Pergamon Press, New York, 1969.

[9] E.S. Santos, "Maximin Automata", *Information and Control* 13 (1968) 363-377.

[10] E.S. Santos, "Realization of Fuzzy Languages by Probabilistic, Max-Product, and Maximin Automata", *Information Sciences* 8 (1975) 39-53.

[11] E.S. Santos, "Regular Fuzzy Expressions Fuzzy Automata", *Fuzzy Automata and Decision Processes, edited by M.M. Gupta, G.N. Saridis and B.R. Gaines*, North-Holland, 1977, 169-175.