Myhill Nerode Theorem for Fuzzy Automata
(Min-max Composition)

V. Ramaswamy\textsuperscript{1} and K. Chatrapathy\textsuperscript{2}

\textsuperscript{1}Bapuji College of Engineering and Technology, VTU-Belgaum, India
Email: researchwork04@yahoo.com

\textsuperscript{2}School of Engineering and Technology, JAIN University, India
Email: kcpathy@yahoo.in

Received 20 December 2013; accepted 28 December 2013

Abstract. In this paper, Myhill Nerode theorem of finite automaton has been extended to fuzzy automata where the composition considered is min-max composition. In the case of max-min composition, it has already been proved that if \( L \) is a fuzzy regular language, then for any \( \alpha \in [0, 1] \), \( L_\alpha = L(D_\alpha(M)) \) [3]. In the case of max-product composition \( L_\alpha \) is only a subset of \( L(D_\alpha(M)) \). But still Myhill Nerode theorem has been extended to max-product composition [4]. In the case of max-average composition, \( L_\alpha \) is not even contained in \( L(D_\alpha(M)) \). This lead to lots of challenges and we had to resort to splitting to prove the analogue of Myhill Nerode Theorem for max-average composition. In a similar line, an attempt has been made in this paper to study the behavior of fuzzy automata under min-max composition and to prove the analogue of Myhill Nerode Theorem for min-max composition. An algorithm to compute \( L(s) \) for any string \( s \) is also developed.

Keywords: Monoid, min-max composition, finite automaton, equivalence class, fuzzy regular language, fuzzy automaton

AMS Mathematics Subject Classification (2010): 68Q45, 68Q70

1. Introduction
Let \( A \) be a finite non empty set. A fuzzy automaton over \( A \) is a 4-tuple \( M = (Q, f, I, F) \) where \( Q \) is a finite nonempty set, \( f \) is a fuzzy subset of \( Q \times A \times Q \). I and F are fuzzy subsets of \( Q \). In other words, \( f: Q \times A \times Q \to [0, 1] \) and I, F: \( Q \to [0, 1] \).

Let \( S \) be a free monoid with identity element \( e \) generated by \( A \). If \( s \in S \), then \( s \) can be written as \( a_1a_2\ldots a_n \) where \( a_i \in A \). Here \( n \) is called the length of \( s \) and we write \( |s| = n \). We now extend \( f \) to a function \( f^*: Q \times S \times Q \to [0, 1] \) defined as

\[
f^*(q, e, p) = \begin{cases} 0 & \text{if } q = p, \\ 1 & \text{otherwise.} \end{cases}
\]

\[
f^*(q, sa, p) = \land [ f^*(q, s, r) \lor f(r, a, p)] \quad (s \in S, a \in A)
\]

\( r \in Q \)

It can be shown that \( f^*(q, a, p) = f(q, a, p) \) for all \( p, q \in Q \) and for all \( a \in A \).
Myhill Nerode Theorem for Fuzzy Automata (Min-max Composition)

Definition 1.1. Let M = (Q, f*, I, F) be a fuzzy automaton over S. We define the language accepted by M denoted by L(M) to be a fuzzy subset of S defined as L(M)(s) = I o f* o F for all s ∈ S. Here o denotes min-max composition.

Definition 1.2. A fuzzy subset L of S is said to be a fuzzy regular language if L = L(M) where M is a fuzzy automaton over S.

2. Myhill Nerode Theorem for Fuzzy Automata

Let S be a monoid with identity element e and L be a fuzzy subset of S. Then the following statements are equivalent.

(i) L is a fuzzy regular language.
(ii) L can be expressed as a fuzzy union

L = (∆1)L ∪ (∆2)L ∪ ... ∪ (∆t)L

where ∆i ∈ [0, 1]. For each i = 1, 2, ..., t, (∆i)L = ∆i.

Lδi

This union is a set theoretic union and [s]δi denotes the equivalence class of s of a right invariant equivalence relation of finite index in Lδi.

(iii) Define a relation R_L as follows.

If s, t ∈ S, then s R_L t if and only if for all u ∈ S and for all α ∈ [0, 1], L(su) ≥ α only when L(tu) ≥ α. Then R_L is a right invariant equivalence relation of finite index.

Proof: (i) → (ii)

Since L is a fuzzy regular language, we have L = L(M) where M = (Q, f*, I, F) is a fuzzy automaton. Consider any α ∈ [0, 1]. With M and α, we associate a non-deterministic automaton Dα(M) = (Q, dα, Iα, Fα) where

dα : Q x S → 2^Q is defined as dα(q, s) = {p ∈ Q | f*(q, s, p) ≥ α}.

Lα = {p ∈ Q | I(p) ≥ α} and
Fα = {p ∈ Q | F(p) ≥ α}.

For the sake of simplicity, we will denote L(Dα(M)) by Lα(M).

Let s ∈ Lα. Then L(s) = L(M)(s) ≥ α, i.e. (I o f* o F) ≥ α which means

∧ [(f* o F)(p) ∨ I(p)] ≥ α

for all p ∈ Q.

This means for any state p ∈ Q, I(p) ≥ α OR (f* o F)(p) ≥ α. This leads to the following three cases:

Case A: I(p) ≥ α and (f* o F)(p) ≥ α

Case B: I(p) < α and (f* o F)(p) ≥ α

Case C: I(p) ≥ α and (f* o F)(p) < α

We now consider each case separately.

Case A: I(p) ≥ α and (f* o F)(p) ≥ α. In this case p ∈ Iα.

Now (f* o F)(p) ≥ α means

∧ [(f*(p, r) ∨ F(r)) ≥ α]

r ∈ Q

This leads to the following three cases:

Case A1: f*(p, r) ≥ α and F(r) ≥ α

Case A2: f*(p, r) ≥ α and F(r) < α

Case A3: f*(p, r) < α and F(r) ≥ α

Case A4: f*(p, r) = f*(p, s, r) ≥ α and F(r) ≥ α

59
V. Ramaswamy and K. Chatrapathy

First alternative means r ∈ \( d_\alpha(p, s) \), \( F(r) ≥ \alpha \) means r ∈ \( F_\alpha \).

Thus r ∈ \( d_\alpha(p, s) ∩ F_\alpha \). Hence \( d_\alpha(p, s) ∩ F_\alpha ≠ \emptyset \) where p ∈ \( I_\alpha \). This proves that s ∈ L(\( D(\alpha(M)) = L_\alpha(M) \)).

**Case A**:

**Case A:** \( f_\alpha^*(p, r) ≥ \alpha \) and F(r) < \( \alpha \)

Let F(r) = \( \beta < \alpha \). Then r ∈ \( F_\beta \). I(p) ≥ \( \alpha > \beta \) means p ∈ \( I_\beta \). Also \( f_\beta^*(p, r) ≥ \alpha > \beta \) means r ∈ \( d_\beta(p, s) \).

Thus r ∈ \( d_\beta(p, s) \) and r ∈ \( F_\beta \) so that \( d_\beta(p, s) ∩ F_\beta ≠ \emptyset \) where p ∈ \( I_\beta \).

This proves that s ∈ L(\( D_\beta(M) = L_\beta(M) \)).

**Case B**:

**Case B:** \( f_\alpha^*(p, r) < \alpha \) and F(r) ≥ \( \alpha \)

Let F(r) = \( \gamma < \alpha \). Then \( f_\gamma^*(p, s, r) = \gamma \) so that r ∈ \( d_\gamma(p, s) \)

\( F_\gamma(r) ≥ \gamma > \gamma \) means r ∈ \( F_\gamma \). I(p) ≥ \( \alpha > \gamma \) means p ∈ \( I_\gamma \).

Thus r ∈ \( d_\gamma(p, s) ∩ F_\gamma \) so that \( d_\gamma(p, s) ∩ F_\gamma ≠ \emptyset \) where p ∈ \( I_\gamma \). This proves that s ∈ L(\( D_\gamma(M) = L_\gamma(M) \)).

**Case B:** \( f_\alpha^*(p, r) ≥ \alpha \) and F(r) < \( \alpha \)

We already have I(p) < \( \alpha \).

Let I(p) = \( \lambda < \alpha \). This implies p ∈ \( I_\lambda \). Now F(r) ≥ \( \alpha > \lambda \) means r ∈ \( F_\lambda \)

and \( f_\lambda^*(p, r) = f_\lambda^*(p, s, r) ≥ \alpha > \lambda \) means r ∈ \( d_\lambda(p, s) \).

Thus r ∈ \( d_\lambda(p, s) ∩ F_\lambda \) so that \( d_\lambda(p, s) ∩ F_\lambda ≠ \emptyset \) where p ∈ \( I_\lambda \). This proves that s ∈ L(\( D_\lambda(M) = L_\lambda(M) \)).

**Case B:** \( f_\alpha^*(p, r) < \alpha \) and F(r) ≥ \( \alpha \)

We already have I(p) < \( \alpha \).

Let I(p) = \( \rho < \alpha \). Then p ∈ \( I_\rho \). Let F(r) = \( \varphi < \alpha \). Then r ∈ \( F_\varphi \).

If \( \rho > \varphi \), then I_\rho ⊆ I_\varphi so that p ∈ \( I_\varphi \). Also \( f_\varphi^*(p, r) = f_\varphi^*(p, s, r) ≥ \alpha > \rho > \varphi \) which means r ∈ \( d_\varphi(p, s) \).

Thus there exists p ∈ \( I_\varphi \) such that \( d_\varphi(p, s) ∩ F_\varphi ≠ \emptyset \). This proves that s ∈ L(\( D_\varphi(M) = L_\varphi(M) \)).

**Case B:** \( f_\alpha^*(p, r) < \alpha \) and F(r) ≥ \( \alpha \)

We already have I(p) < \( \alpha \).

Let I(p) = \( \pi < \alpha \). Then p ∈ \( I_\pi \) and F(r) ≥ \( \alpha > \pi \) implies r ∈ \( F_\pi \).

Let \( f_\pi^*(p, r) = f_\pi^*(p, s, r) = \mu < \alpha \).

If \( \mu ≤ \pi \), then F(r) ≥ \( \alpha > \mu \) implies r ∈ \( F_\mu \) and \( f_\mu^*(p, s, r) = \mu \) means r ∈ \( d_\mu(p, s) \).

Also \( I(p) = \pi ≥ \mu \) means p ∈ \( I_\mu \) so that \( d_\mu(p, s) ∩ F_\mu ≠ \emptyset \) where p ∈ \( I_\mu \). This proves that s ∈ L(\( D_\mu(M) = L_\mu(M) \)).

If \( \mu > \pi \), then \( f_\mu^*(p, s, r) = \mu > \pi \) means r ∈ \( d_\mu(p, s) \). Thus r ∈ \( d_\mu(p, s) ∩ F_\mu ≠ \emptyset \) so that \( d_\mu(p, s) ∩ F_\mu ≠ \emptyset \).

This proves that s ∈ L(\( D_\mu(M) = L_\mu(M) \)).

**Case C:** I(p) ≥ \( \alpha \) and \( f_\alpha^*(\varphi, F(p)) < \alpha \).

This implies \( f_\alpha^*(p, r) < \alpha \) and F(r) < \( \alpha \).
Myhill Nerode Theorem for Fuzzy Automata (Min-max Composition)

We already have I (p) ≥ α.

Let f*(p, r) = v < α and F (r) = Ω < α.

First assume that v > Ω. Now F (r) = Ω means r ∈ FΩ. Also
f*(p, s, r) = v ≥ Ω means r ∈ dΩ(p, s). Hence dΩ(p, s) ∩ FΩ ≠ ∅. Also
I (p) ≥ v > Ω means p ∈ IΩ. Hence
s ∈ L (δΩ(M)) = LΩ(M). (9)

Suppose v < Ω. Now F (r) = Ω > v means r ∈ Fv. f*(p, r, r) = v means
r ∈ dδ(p, s) so that dδ(p, s) ∩ Fv ≠ ∅. Also I (p) ≥ v > Ω means p ∈ Iv.

Hence s ∈ L (Dδ(M)) = Ls(M). (10)

From (1), (2), (3), (4), (5), (6), (7), (8), (9), and (10) it follows that
Lα = Lα(M) ∪ Lδ(M) ∪ Lα(M) ∩ Lδ(M) ∪ Lα(M) ∩ Lδ(M) ∪ Lα(M) ∩ Lδ(M) ∩ Lδ(M) ∪ Lα(M) ∩ Lδ(M) ∩ Lδ(M) where α, β, γ, δ, ε, ζ, ξ, η ∈ {0, 1}.

Since each of the languages Lα(M), Lδ(M), Ls(M), …, Lη(M) are fuzzy regular languages accepted by non-deterministic automata Dα(M), Dδ(M), Ds(M), …, Dη(M) respectively, Myhill Nerode theorem for finite automata is applicable for each automaton. Let Q = {q0, q1, q2, …, qn}. For every s ∈ S, the possible values of L(s) are I(q0), I(q1),…, I(qn), f(qi, a1, qj) (qi, qj ∈ Q, ai ∈ A). F(qj), F(qj)…, F(qj). Denote these fixed values (after arranging them in non decreasing order) by δ1, δ2,…, δn. So, there can be only finitely many values of L(s) (s ∈ S). Then δi, δi+1, δi ∈ [0, 1] and for each δi (1 ≤ i ≤ n), Lα = Lα(M) ∩ Lδ(M) ∩ Ls(M) ∩ Lη(M) ∩ Lα(M) ∩ Lδ(M) ∩ Ls(M) ∩ Lη(M)

Since L (Dα(M)) is the language accepted by a finite automaton, by Myhill Nerode theorem for finite automata, it follows that there exists a right invariant equivalence relation R, of finite index. Let R′ denotes its restriction on Lα. Similarly, we obtain other restrictions like M0, N0, O0, P0, Q0, S0, T0, U0, V0, W0, X0, and Y0 from L (Dα(M)), L (Dδ(M)), L (Dη(M)), L (Dη(M)), L (Dα(M)), L (Dη(M)), L (Dη(M)), L (Dη(M)) respectively. Note that M0, N0, O0, P0, Q0, S0, T0, U0, V0, W0, X0, and Y0 are all right invariant equivalence relations of finite index. Hence Z0 = M0 ∩ N0 ∩ O0 ∩ P0 ∩ Q0 ∩ S0 ∩ T0 ∩ U0 ∩ V0 ∩ W0 ∩ X0 ∩ Y0 is a right invariant equivalence relation in Lα of finite index. Let [s]α denote the equivalence class of S under this equivalence relation. Since the equivalence classes partition Lα, it follows that Lα = ∪ [s]α.

Next we will prove the fact that L = (δ1) L ∪ (δ2) L ∪ … ∪ (δn) L.

Define (δi) L = δi . Lα. If s ∈ S such that L(s) ≥ δi (s ∈ Lα), then (δi) L(s) = δi.

Otherwise, (δi) L(s) = 0. We note that each (δi) L is a fuzzy set. Let s ∈ S and assume that L(s) = δi. Now L(s) = δi ≤ δi+1 ≤ … ≤ δi. Again, L(s) = δi ≥ δi-1 ≥ … ≥ δi.

Hence (δi) L ∪ (δi) L ∪ … ∪ (δi) L(s) = (δi) L(s) ∨ (δi) L(s) ∨ … ∨ (δi) L(s) = δi ∨ δi ∨ … ∨ δi = δi = L (s). This proves that L = (δ1) L ∪ (δ2) L ∪ … ∪ (δn) L.

Proof: (ii) → (iii).

If s ∈ S, then s R L s because for all u ∈ S and for all α ∈ [0, 1], L(su) ≥ α only when L(su) ≥ α is obviously true. This proves that R L is reflexive. Clearly, R L is symmetric. If s R L t and t R L v, then for all u ∈ S and for all α ∈ [0, 1], L(su) ≥ α only when L(tu) ≥ α only
V. Ramaswamy and K. Chatrapathy

when \( L(vu) \geq \alpha \) proving that \( s R_L t \). Hence \( R_L \) is transitive. \( R_L \) is thus an equivalence relation.

To prove \( R_L \) is right invariant, assume that \( s R_L t \) and \( u \in S \). We have to prove that \( su \in R_L tu \). For this, we have to prove that for all \( v \in S \) and \( \alpha \in [0, 1] \), \( L(suv) \geq \alpha \) only when \( L(tuv) \geq \alpha \) which is the same as saying that \( L(sz) \geq \alpha \) only when \( L(tz) \geq \alpha \) where \( z = uv \). But this is true since \( s R_L t \).

We will now prove that \( R_L \) is of finite index. For \( i = 1, 2, \ldots, t \), let \( R_i \) denote the right invariant equivalence relation of finite index in \( L_{\delta_i} \). Let \( R = R_1 \cap R_2 \cap \ldots \cap R_t \). Then \( R \) is an equivalence relation of finite index. We will prove that \( s R t \) implies \( s R_L t \). This will mean that index \( (R_L) \leq \) index \( (R) \). Since index \( (R) \) is finite, this will prove that index \( (R_L) \) is also finite.

Assume that \( s R t \). Consider any \( u \in S \) and any \( \alpha \in [0, 1] \). Suppose \( su \in L_\alpha \). We have to prove that \( tu \in L_\alpha \). Now \( \alpha \leq L(su) = \delta_j \) (say). Then \( su \in L_{\delta_j} \) which is a subset of \( L_\alpha \). By definition of \( R \), we have \( s R_j t \). Since \( R_j \) is right invariant, \( su R_j tu \). Since \( L_{\delta_j} = U \{v \mid \delta_j \} \), it follows that \( su \) belongs to one of the equivalence classes of \( R_j \) and hence \( tu \) also belongs to the same equivalence class. Hence \( tu \in L_{\delta_j} \) and since \( L_{\delta_j} \) is a subset of \( L_\alpha \), we have \( tu \in L_\alpha \).

**Proof: (iii) → (i)**

We have to define a fuzzy automaton \( M \) such that \( L = L(M) \). For every element \( s \in S \), let \([s]\) denote the equivalence class of \( s \) under the equivalence relation \( R_L \).

Let \( Q = \{ [s] \mid s \in S \} \). Since \( R_L \) is of finite index, it follows that \( Q \) is a finite set. Define \( I: Q \rightarrow [0, 1] \), \( f^* : Q \times S \times Q \rightarrow [0, 1] \) and \( F : Q \rightarrow [0, 1] \) as follows.

\[ I([s]) = 0 \text{ if } [s] = [e] = 1 \text{ otherwise.} \]

\[ f^*([s], t, [u]) = 1 \text{ if } [u] = [st], 0 \text{ otherwise.} \]

\[ F([s]) = L(s). \]

We will first prove that \( F \) is well defined. For this, we have to prove that if \([s] = [t]\), then \( L(s) = L(t) \). Assume that \( L(s) = \beta \). We will prove that \( L(t) = \beta \). Since \([s] = [t]\), \( s R_L t \) so that \( L(s) = L(se) \geq \beta \) only when \( L(t) = L(te) \geq \beta \). Since \( L(s) \geq \beta \), it follows that \( L(t) \geq \beta \).

Assume \( L[t] = \gamma > \beta \). Take \( \eta = (\beta + \gamma) / 2 \). Clearly, \( \beta < \eta < \gamma = L[t] \). Since \( s R_L t \), \( L[t] > \eta \) implies that \( L[s] \geq \eta > \beta \). But this contradicts the fact that \( L(s) = \beta \). Hence our assumption that \( L[t] > \beta \) is wrong. Since \( L[t] \geq \beta \), it follows that \( L[t] = \beta \).

Take \( M = (Q, I, f^*, F) \). Then \( M \) is a fuzzy automaton and it remains to prove that \( L = L(M) \).

For this, we have to prove that for all \( s \in S \), \( L(s) = L(M)(s) \).

We have

\[ L(M)(s) = I \circ f^*_s \circ F \]
\[ = \land \{ I([t]) \lor (f^*_s \circ F)([t]) \} \]
\[ = \land \{ f^*_s ([t], [u]) \lor F([u]) \} \]
\[ = \land \{ f^*_s ([t], s, [u]) \lor F([u]) \} \]

Note that \( f^*_s ([t], s, [u]) = 0 \) if \([ts] = [u]\) and 1 otherwise. Therefore, in the above expression the term \( f^*_s ([t], s, [u]) \) becomes 1. Thus the above equation becomes...
Myhill Nerode Theorem for Fuzzy Automata (Min-max Composition)

\[(f^*, o F) ([t]) = F([u]) \]

\[= L(ts) \quad (\text{since } F([s]) = L(s)). \]

Hence \( L(M)(s) = \bigwedge \{I([t]) \lor (f^*, o F) ([t])\} \)

\[\text{[t]} \]

Note that \(I([t]) = 0\) only when \([t] = [e]\). \(I([t]) = 1\) whenever \([t] \neq [e]\). Therefore, \(\{I([t]) \lor (f^*, o F) ([t])\} = 1\) whenever \([t] \neq [e]\) and \(\{I([t]) \lor (f^*, o F) ([t])\} = (f^*, o F) ([t])\) when \([t] = [e]\). Thus the above equation becomes

\(L(M)(s) = (f^*, o F) ([t])\) where \([t] = [e]\).

\[= L(ts) \quad (\text{by the above result}) \]

\[= L(es) \quad (\text{since } I[t] = 0 \text{ when } [t]=[e] \text{ and } R_L \text{ is a right invariant relation, } [ts] = [es] ) \]

\[= L(s) \]

Thus for all \(s\) all \(s \in S\), \(L(s) = L(M)(s)\). This proves that \(L = L(M)\).

3. Implementation

The algorithm to compute \(L(s) = L(M)(s)\) for any string \(s\) of arbitrary length and any fuzzy automata \(M\) with any number of states is developed and implemented in C++. Following procedures are used to compute \(f^*(q_i, s, q_j)\) and \(L(s)\) for all \(s \in S\) and \(q_i, q_j \in Q\).

**Procedure MinMax(i,j,X,Y).** This procedure computes and returns the min-max composition value of row-I of matrix \(X\) and column-j of matrix \(Y\). \(X\) and \(Y\) are the \(n \times n\) transition matrices, \(min\), \(temp\) and \(r\) are temporary variables.

1. \(min = \infty\)
2. for \(r = 0\) to \(n-1\) do
   2.1 if \((X[i][r] \geq Y[r][j])\) then
      \(temp = X[i][j]\)
   else
      \(temp = Y[i][j]\)
   2.2 if \((min > temp)\) then
      \(min = temp\)
3. return \(min\)

**Procedure computeFstar (s).** This procedure computes \(f^*\) - matrix for the input string \(s\) and stores it in \(n \times n\) matrix \(A\). \(F0\) and \(F1\) are the transition matrices for the input symbols 0 and 1 respectively. The procedure call \(COPY(X, Y)\) copies the matrix \(X\) to matrix \(Y\). \(B\) is the temporary matrix of size \(n \times n\). The procedure call \(computeFstar(X, Y, Z)\) computes the \(f^*\)-value for each pair \((q_i, q_j) \in Q \times Q\) using transition matrices \(X\), \(Y\) and stores the result in the matrix \(Z\).

1. if \((s[0] = '0')\) then
   \(COPY(A, F0)\)
else
   \(COPY(A, F1)\)
2. for \(i = 1\) to \((\text{length}(s) - 1)\) do
   if \((s[i] = '0')\)
      \(computeFstar(A, F0, B)\)
   else
      \(computeFstar(A, F1, B)\)
V. Ramaswamy and K. Chatrapathy

Procedure computeFstarCompF(q). This procedure computes and returns $(f^*, o F)(q)$ value for a given state $q \in Q$. $A$ is the $f^*$ matrix for the string $s$.

1. $\min = \infty$
2. for $r = 0$ to $n-1$ do
   2.1 temp = $\max(A[p][r], F[r])$
   2.2 if (temp < min) then
       min = temp
3. return min

Procedure computeLs. This procedure computes and returns $L(s)$ value for a given string $s$.

1. $\min = \infty$
2. for $p = 0$ to $n-1$ do
   2.1 temp = computeFstarCompF(p)
   2.2 if ($I[p] > temp$) then
       temp = $I[p]$
   2.3 if (temp < min) then
       min = temp
3. return min

Procedure main(). This procedure inputs the fuzzy automaton $M = (Q, f, I, F)$, computes and returns $L(s)$ value for a given input string $s$. $F_0$, $F_1$, $n$ are transition matrix for 0, transition matrix 1 and number of states in $Q$ respectively. $F_e$ is the $f^*$-matrix for e. $I$ and $F$ are array of size $n$. $L_s$ stores the $L(s)$ value of the input string $s$.

1. read number of states $n$
2. read arrays $I$ and $F$
3. set $f^*_e$-matrix $F_e$
4. read transition matrices $F_0$, $F_1$
5. $ch = \text{‘y’}$
6. while ($ch = \text{‘y’}$) do
   6.1 Read input string $s$
   6.2 $A = \text{computeFstar}(s)$
   6.3 $L_s = \text{computeLs}( )$
   6.4 Print transition matrix $A$
   6.5 Print $L_s$
   6.6 read input character $ch = \text{‘y’}$ to continue, $ch = \text{‘n’}$ to stop
7. Exit

The program is tested for large number of fuzzy automata and strings of arbitrary length.

4. Example
Myhill Nerode Theorem for Fuzzy Automata (Min-max Composition)

Let $\Sigma = \{0, 1\}$ and $S = \Sigma^*$, the set of all strings over the alphabet $\Sigma$. Consider the fuzzy automaton $M = (Q, f, I, F)$ where $Q = \{q_0, q_1, q_2\}$, $f$ is the fuzzy subset $f: Q \times \Sigma \times Q \rightarrow [0, 1]$ defined as

- $f(q_0, 0, q_0) = 0.0$, $f(q_0, 0, q_1) = 0.8$, $f(q_0, 0, q_2) = 0.6$
- $f(q_1, 0, q_0) = 0.5$, $f(q_1, 0, q_1) = 0.0$, $f(q_1, 0, q_2) = 0.7$
- $f(q_2, 0, q_0) = 0.3$, $f(q_2, 0, q_1) = 0.6$, $f(q_2, 0, q_2) = 0.0$
- $f(q_0, 1, q_0) = 0.0$, $f(q_0, 1, q_1) = 0.6$, $f(q_0, 1, q_2) = 0.7$
- $f(q_1, 1, q_0) = 0.5$, $f(q_1, 1, q_1) = 0.0$, $f(q_1, 1, q_2) = 0.8$
- $f(q_2, 1, q_0) = 0.4$, $f(q_2, 1, q_1) = 0.2$, $f(q_2, 1, q_2) = 0.0$

$I = \{q_0\}$ and $F$ is the fuzzy subset of $Q$ defined as $F(q_1) = 0.4$ and $F(q_2) = 0.9$.

For any string $w = sa$ of length two or more we will calculate $f^*(q, sa)$ as follows:

$$f^*(q, sa) = \bigwedge_{r \in Q} [f^*(q, s, r) \lor f(r, a, p)] \quad (s \in S, a \in \Sigma, q_0, q_1 \in Q)$$

After computing $f^*$-matrix for a given string $s$, we will compute $L(M)(s)$ as follows:

$$L(M)(s) = I \circ f^* \circ F$$

Therefore, for any string $s \in S$,

$$L(M)(s) = (f^* \circ o F)(q_1) \land (f^* \circ o F)(q_2)$$

(11)

$$L(0) = L(M)(0) = I \circ f_0^* \circ F$$

(12)

Similarly, $L(1) = 0.4$
V. Ramaswamy and K. Chatrapathy

\[ f^{0*} (q_0, q_1) = f^* (q_0, 00, q_0) = \left[ f (q_0, 0, q_0) \lor f (q_0, 0, q_1) \right] \land \left[ f (q_0, 0, q_1) \lor f (q_1, 0, q_1) \right] \land \left[ f (q_0, 0, q_2) \lor f (q_2, 0, q_1) \right] = 0.6 \]

Similarly, the \( f^{00*} \) matrix is computed as follows:

\[
\begin{align*}
\text{L}(00) &= (f_{00}^* \circ F)(q_1) \land (f_{00}^* \circ F)(q_2) \\
&= \{ 0.4 \land f_{00}^*(q_0, q_0) \land \left[ 0.9 \lor f_{00}^*(q_0, q_1) \right] \} \lland \{ 0.4 \land f_{00}^*(q_1, q_0) \land \left[ 0.9 \lor f_{00}^*(q_1, q_1) \right] \} = 0.3
\end{align*}
\]

Using the program for the example fuzzy automata, \( f^*_s \) matrix and \( L(s) \) values are computed for various strings and the same values are checked using manual calculations. Both manually calculated values and computer results are tallied. Some of the \( L(s) \) values are as follows.

\[
\begin{align*}
L(0) &= 0.3, L(1) = 0.4, L(00) = L(01) = L(10) = 0.3, L(11) = 0.4, L(000) = L(001) = \ldots, L(110) = 0.3, L(111) = 0.4, \\
L(0000) = \ldots, L(1110) = 0.3, L(1111) = 0.4, \\
L(10000000) = 0.3, L(00011111) = 0.3, L(11110000) = 0.3, L(01011100) = 0.3, \\
L(1101011011) = 0.3, L(1111111111) = 0.4, \\
L(1111111111111111) = 0.4.
\end{align*}
\]

It is found that \( L(s) = 0.4 \) only when every symbol in \( s \) is 1. Otherwise, \( L(s) = 0.3 \).

The possible values of \( \delta_i \) (after arranging them in nondecreasing order) are 0.3, 0.4.

Suppose 0 < \( \alpha \leq 0.3 \).

Let \( D_\alpha(M) = M_\alpha \) denote the nondeterministic automaton corresponding to \( \alpha \).

Then \( I_\alpha = \{ q_0 \}, F_\alpha = \{ q_1, q_2 \}, d_\alpha(q_0, s) = \{ p \in Q / f^*_\alpha(q_0, s) \geq 0.3 \} = \{ q_1, q_2 \} \)

\[
\begin{align*}
L(D_\alpha(M)) &= \{ s \in S / \text{there exists } q \in I_\alpha \text{ such that } (d_\alpha(q, s) \cap F_\alpha) \neq \emptyset \} \\
&= \{ s \in S / \text{there exists } q \in I_{0.3} \text{ such that } (d_{0.3}(q, s) \cap F_{0.3}) \neq \emptyset \} = \{ 0, 1 \}^* \\
L_\alpha &= \{ s \in S / L(s) \geq \alpha \} \\
&= \{ s \in S / L(s) \geq 0.3 \} = \{ 0, 1 \}^* \\
L(D_\alpha(M)) &= L_\alpha
\end{align*}
\]

Furthermore, \( [0]_\alpha = \{ 00, 01, 10, 000, 001, 010, 110, 0000, 1110, 00000, \ldots \} \)

\[
\begin{align*}
[1]_\alpha &= \{ 1, 11, 111, 1111, \ldots \} \\
L_\alpha &= \cup [s]_\alpha = [0]_\alpha \cup [1]_\alpha
\end{align*}
\]

Suppose 0.3 < \( \alpha \leq 0.4 \).

Let \( D_\alpha(M) = M_\alpha \) denote the nondeterministic automaton corresponding to \( \alpha \).

Then \( I_\alpha = \{ q_0 \}, F_\alpha = \{ q_1, q_2 \}, d_\alpha(q_0, s) = \{ p \in Q / f^*_\alpha(q_0, s) \geq 0.4 \} = \{ q_1, q_2 \} \)

\[
\begin{align*}
L(D_\alpha(M)) &= \{ s \in S / \text{there exists } q \in I_\alpha \text{ such that } (d_\alpha(q, s) \cap F_\alpha) \neq \emptyset \} \\
&= \{ s \in S / \text{there exists } q \in I_{0.4} \text{ such that } (d_{0.4}(q, s) \cap F_{0.4}) \neq \emptyset \} = \{ 0, 1 \}^* \\
L_\alpha &= \{ s \in S / L(s) \geq \alpha \}
\end{align*}
\]
Myhill Nerode Theorem for Fuzzy Automata (Min-max Composition)

\[ \{ s \in S \mid L(s) \geq 0.4 \} = \{ 1 \} \]

\[ L(D_\alpha(M)) \neq L_\alpha \text{ and also } L_\alpha \subseteq L(D_\alpha(M)). \]

Furthermore, \[ [1]_\alpha = \{ 1, 11, 111, 1111 \ldots \} \]

If \( \alpha > 0.4 \), then there exists no corresponding nondeterministic automaton and \( L(D_\alpha(M)) = L_\alpha = \emptyset \).

When \( \alpha = 0.3 \)

\[ \alpha_L(s) = \alpha \text{ if } L(s) \geq \alpha, \text{ 0 otherwise.} \]

\[ \alpha_L(0) = \alpha_L(00) = \alpha_L(01) = \alpha_L(10) = \alpha_L(000) = \ldots \alpha_L(110) = \alpha_L(0000) = \ldots \alpha_L(111) = \ldots = 0.3 \]

\[ \alpha_L(1) = \alpha_L(11) = \alpha_L(111) = \alpha_L(1111) = \ldots \alpha_L(111111\ldots111) = 0 \]

When \( \alpha = 0.4 \)

\[ \alpha_L(s) = \alpha \text{ if } L(s) \geq \alpha, \text{ 0 otherwise.} \]

\[ \alpha_L(0) = \alpha_L(00) = \alpha_L(01) = \alpha_L(10) = \alpha_L(000) = \ldots \alpha_L(110) = \alpha_L(0000) = \ldots \alpha_L(1110) = 0 \]

\[ \alpha_L(1) = \alpha_L(11) = \alpha_L(111) = \alpha_L(1111) = \ldots \alpha_L(111111\ldots111) = 0.4 \]

\[ L = \alpha_L \text{ where } \cup \text{ denotes fuzzy union.} \]

\[ \alpha \in [0, 1] \]

\[ (\cup \alpha_L)(0) = \vee \alpha_L(0) = 0.3 \vee 0 = 0.3 = L(0) \]

\[ (\cup \alpha_L)(1) = \vee \alpha_L(1) = 0 \vee 0.4 = 0.4 = L(1) \]

Similarly, \( (\cup \alpha_L)(s) = \vee \alpha_L(s) = 0.3 \vee 0 = 0.3 = L(s) \) for all \( s \in S \).

This verifies \( L = \cup \alpha_L \).

5. Results and Conclusions

In this paper, Myhill Nerode theorem of finite automaton has been extended to fuzzy automaton where the composition considered is min-max composition. The algorithm to compute \( f^*(q_i, s, q_j) \) and \( L(s) \) is developed and implemented in C++. The program is tested with different fuzzy automata and strings of different lengths. In min-max composition, it is found that \( L_\alpha \) need not even be contained in \( L(D_\alpha(M)) \). Anyway, we have been able to prove the analogue of Myhill Nerode Theorem for fuzzy automata even for min-max composition.

REFERENCES