Research Article

The Lattice-Valued Turing Machines and the Lattice-Valued Type 0 Grammars

Juan Tang, 1 Yong Fang, 1 and Jian-Gang Tang 2

1 College of Electronics and Information Engineering, Sichuan University, Chengdu, Sichuan 610064, China
2 College of Mathematics and Statistics, Yili Normal University, Yining, Xinjiang 835000, China

Correspondence should be addressed to Yong Fang; yfang2013@qq.com

Received 25 December 2013; Accepted 10 April 2014; Published 8 May 2014

Academic Editor: Yang Xu

Copyright © 2014 Juan Tang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Purpose. The purpose of this paper is to study a class of the natural languages called the lattice-valued phrase structure languages, which can be generated by the lattice-valued type 0 grammars and recognized by the lattice-valued Turing machines. Design/Methodology/Approach. From the characteristic of natural language, this paper puts forward a new concept of the l-valued Turing machine. It can be used to characterize recognition, natural language processing, and dynamic characteristics. Findings. The mechanisms of both the generation of grammars for the lattice-valued type 0 grammar and the dynamic transformation of the lattice-valued Turing machines were given. Originality/Value. This paper gives a new approach to study a class of natural languages by using lattice-valued logic theory.

1. Introduction

According to Chomsky’s rationalist theory, the language can be divided into types by the grammar, and then, types can be recognized and processed, respectively. The fuzziness is the typical trait of the natural languages, and then, how to recognize and deal with the fuzzy natural language is becoming the more and more important subject. From the point of the reorganization for the natural language, in 1967, Wee [1] firstly introduced the fuzzy finite automata, which generated many interesting explorations. Its important applications in learning systems, automatic control, pattern recognition and database, and so on were also studied by many researchers; the more details and the fuzzy finite automata were studied by Wee and Fu [2], Santos [3–6], Lee and Zadeh [7], Kumbhojkar and Chaudhari [8, 9], Malik et al. [10–15], and so on; the authors can also refer to [16, 17].

In this paper, our main purpose is to consider a class of the natural languages which is called the lattice-valued phrase structure languages. In fact, these natural languages can be generated by the lattice-valued type 0 grammars and recognized by the lattice-valued Turing machines. Moreover, the natural languages can be described by the formation mechanism and the transferred mechanism of the grammar.

This paper is organized as follows. Some basic concepts such as the complete Heyting algebra, the lattice-valued language, and the corresponding membership were recalled in Section 2. In Section 3, we mainly studied the lattice-valued Turing machine and the lattice-valued recursively enumerable language. Especially, the mechanisms of both the generation of grammars for the lattice-valued type 0 grammar and the dynamic transformation of the lattice-valued Turing machines were given; we used the lattice-valued logic theory to study the class of the natural languages; it seems to be that the approach which we used is new as far as we know. Finally, we study the equivalence between the lattice-valued type 0 grammar and the lattice-valued Turing machine in Section 4.

2. Preliminaries

Throughout this paper, a nonempty finite set $\Sigma$ is called an alphabet. The element in the alphabet is called a symbol or a letter. And, $\Sigma^*$ denotes the free monoid, that is, the set of all
strings with letters from $\Sigma$ and the empty string, where the empty string is denoted by $\epsilon$; moreover, the length of a string $x$ is denoted by $|x|$, which is the number of symbols in the string; for the empty string $\epsilon$, we set $|\epsilon| = 0$.

Given an alphabet $\Sigma$. The concatenation of two strings $x$ and $y$ is the string which was denoted by $xy$ and was obtained by appending the symbols of $y$ to the right end of $x$. Assume $x$ is a string, and then $x^n$ stands for the string obtained by repeating the string $x$ by $n$ times. As a special case, we can define $x^0 = \epsilon$.

Let $\Sigma_1, \Sigma_2$ be two alphabets; the product of $\Sigma_1$ and $\Sigma_2$ is given by

$$\Sigma_1 \Sigma_2 = \{ab \mid a \in \Sigma_1, b \in \Sigma_2\}. \quad (1)$$

Define $n$ power of an alphabet $\Sigma$ by

$$\Sigma^0 = \epsilon, \quad \Sigma^n = \Sigma^{n-1} \Sigma, \quad n \geq 1. \quad (2)$$

The positive closure of $\Sigma$ is

$$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \Sigma^4 \ldots, \quad (3)$$

and the star-closure of $\Sigma$ is

$$\Sigma^* = \Sigma^0 \cup \Sigma^+ = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \Sigma^4 \ldots. \quad (4)$$

Let $\Gamma$ be a complete Heyting algebra and $\Sigma$ an alphabet. We call a map $\varphi : \Sigma^* \rightarrow \Gamma$ a lattice-valued language on $\Sigma$ which is defined by

$$\varphi(\varnothing) = 0, \quad \varphi(\epsilon) = 1 \text{ and } \varphi(ab) = \varphi(a) \lor \varphi(b). \quad (5)$$

Then $\mathcal{L}$ is called a complete lattice-valued language.

Let $\Gamma$ be a complete Heyting algebra and $\Sigma$ an alphabet. We call a map $\varphi : \Sigma^* \rightarrow \Gamma$ a lattice-valued language. $\forall x \in \Sigma^*$, $\varphi(x)$ denotes the membership degree of $x$ belonging to the lattice-valued language.

Let $\varphi, \varphi_1,$ and $\varphi_2$ be the lattice-valued languages on $\Sigma$. Then

$$\varphi_1 \lor \varphi_2$$

is a lattice-valued language on $\Sigma$ via

$$(\varphi_1 \lor \varphi_2)(x) = \varphi_1(x) \lor \varphi_2(x) \quad (6)$$

and the positive closure, the star-closure of $\Sigma$, then

$$\varphi_1 \land \varphi_2 = \varphi_1(\varphi_2 \lor \cdots \lor \varphi_n) \quad (7)$$

$$(\varphi_1 \land \varphi_2)(x) = \varphi_1(x) \land \varphi_2(x).$$

Generally, if $\varphi_1, \varphi_2, \ldots, \varphi_n$ are the lattice-valued languages on $\Sigma$, then

$$\bigvee_{i=1}^n \varphi_i = \varphi_1 \lor \varphi_2 \lor \cdots \lor \varphi_n \quad (8)$$

and the star-closure of $\Sigma$, then

$$(\varphi_1 \lor \cdots \lor \varphi_n)(x) = \varphi_1(x) \lor \cdots \lor \varphi_n(x).$$

Let $\varphi_1$ and $\varphi_2$ be lattice-valued languages on $\Sigma$ which is defined by

$$\varphi_1 \circ \varphi_2(x) = \bigvee \{ \varphi_1(y) \land \varphi_2(z) \mid x = yz \}. \quad (9)$$

(3) Let $\theta \in \mathcal{L}$; then $\theta \varphi$ is a lattice-valued language on $\Sigma$ which is defined by

$$(\theta \varphi)(x) = \theta \land \varphi(x). \quad (10)$$

(4) $\varphi^*$ is a lattice-valued language on $\Sigma$ which is defined by

$$\varphi^*(x) = \begin{cases} \varphi(\epsilon) \land \varphi(b_1) \land \cdots \land \varphi(b_n), & \text{if } x = \epsilon \\ \lor \{ \varphi(b_1) \land \varphi(b_2), \ldots, \varphi(b_n) \} & \text{if } x \neq \epsilon. \end{cases} \quad (11)$$

Let $X$ be a nonempty set and $L$ be a complete Heyting algebra. The map

$$R : X \times X \rightarrow L \quad (12)$$

is called the binary lattice-valued relation on $X$. If $R(x, y) = \theta$, we denote it by $xR_y$, which can be understood as the membership degree $\theta$ of $x$ and $y$ satisfying the relation $R$.

Let $R, S$ be binary lattice-valued relations on $X$; the composition operation of $R$ and $S$ is defined by

$$(R \circ S)(x, z) = \bigvee_{y \in X} (R(x, y) \land S(y, z)). \quad (13)$$

Consider the following:

(a) $R^0(x) = \begin{cases} 1 & x = y \\ 0 & x \neq y \end{cases} \quad (14)$

(b) $R \subseteq S \Rightarrow R(x, y) \leq S(x, y), \forall (x, y) \in X \times X;

(c) $(R \cup S)(x, y) = R(x, y) \lor S(x, y), (R \cap S)(x, y) = R(x, y) \land S(x, y);

(d) $R^n = (R^{-1} \circ R)^n$, $R^* = \bigcup_{n=0}^{\infty} R^n$ are called $n$ power, the positive closure, and the star-closure of $R$, respectively.

3. The Lattice-Valued Turing Machine and the Lattice-Valued Recursively Enumerable Language

A lattice-valued Turing machine $\mathcal{M}$ is a 7-tuple

$$\mathcal{M} = (Q, \Gamma, \Sigma, \delta, q_0, B, F), \quad (15)$$

where

$Q$ is a finite set of states;

$\Gamma$ is a finite set of tape symbol and $B$ is a special symbol of $\Gamma$ called the blank;

$\Sigma$ is a subset of $\Gamma$ not containing $B$ and is called the input symbols set;
\( \delta \) is a transition function, that is, a map from \( Q \times \Gamma \) to \( Q \times \Gamma \times \{-1, +1\} \times \mathcal{L} \):

\[ \delta(q, X) = (p, Y, +1, \theta) \]

\( \theta \) represents the membership degree of \( Y \) in place of \( X \) in the state \( q \), the next state is \( p \), and the read-write head moves right one unit after the symbol \( Y \) in place of \( X \) on the tape. \( \mathcal{L} \) is the set of all symbols on the input tape. 

The above description is the definition of a single tape lattice-valued Turing machine. For the \( n \) tapes lattice-valued Turing machine, the transition function is defined as the map from \( Q \times \Gamma^n \) to \( Q \times \Gamma^n \times \{-1, +1\} \times \mathcal{L} ^n \).

We use \( \alpha_1, q_0 \alpha_2 \) to denote an instantaneous description (ID for short) of the lattice-valued Turing machine \( \mathcal{M} \), where \( q_0 \in Q \) is the current state of \( \mathcal{M} \) and \( \alpha_1, \alpha_2 \) is the string of \( \Gamma^n \).

The membership degree of \( Y \) replacing \( X \) is \( \theta \); that is,

\[ \vdash (ID_1, ID_2) = \theta \begin{cases} \vdash ID_1 & \text{or} \vdash ID_2 \end{cases} \]

If \( \delta(q, X) = (p, Y, +1, \theta) \), then if \( i \neq 1 \), the next ID of \( \mathcal{M} \) is

\[ ID_2 = X_1 X_2 \cdots X_{i-1} p X_i X_{i+1} \cdots X_n \]

If \( i = 1 \), before \( \mathcal{M} \) moves on, the read-write head has been at the far left of the input tape, then the read-write head moves left, which would make the read-write head away from the input tape, which is not allowed. In order to avoid this phenomenon, in this case, we have defined that \( \mathcal{M} \) has not the next ID.

\[ \vdash (ID_1, ID_2) = \theta \begin{cases} \vdash ID_1 & \text{or} \vdash ID_2 \end{cases} \]

\[ \vdash X_1 X_2 \cdots p X_{i-1} X_{i+1} \cdots X_n \]

\[ \vdash \theta \begin{cases} \vdash X_1 X_2 \cdots p X_{i-1} X_{i+1} \cdots X_n \end{cases} \]

Obviously, \( \vdash \_\_ \) is a binary lattice-valued relation on \( \Gamma^n \times \Gamma^n \times \Gamma^+ \).

\[ \vdash \_\_ \] denotes the positive closure of \( \vdash \_\_ \); that is, \( \vdash \_\_ \) = \( \vdash \_\_ \) + \( \vdash \_\_ \) + \( \vdash \_\_ \) + \( \vdash \_\_ \).

\[ \vdash \_\_ \] denotes the star-closure of \( \vdash \_\_ \); that is, \( \vdash \_\_ \) = \( \vdash \_\_ \).

Let \( ID_1, ID_2 \in \Gamma^n \times \Gamma^n \times \Gamma^+ \). By the composition definition of the binary lattice-valued relations, it is not difficult to see the following:

\[ \vdash \_\_ \] = \( \vdash \_\_ \) \( \vdash \_\_ \) \( \vdash \_\_ \) \( \vdash \_\_ \).

**4. The Lattice-Valued Type 0 Grammar and Lattice-Valued Phrase Structure Language**

The lattice-valued grammar \( \mathcal{G} \) is a 4-tuple \( \mathcal{G} = (V, T, P, S) \), where we have the following:

- \( V \) is a nonempty finite set of variables and \( \forall A \in V, A \) is called a syntactic variable (variable for short) or a nonterminal symbol. It represents a syntactic category.

- \( T \) is a nonempty finite set of the terminal symbols and \( \forall a \in T, a \) is called a terminal symbol. Since the variables in \( V \) represent the syntactic category and the characters in \( T \) are the characters that appear in the sentence of language, so we have \( V \cap T = \emptyset \).

- \( P \) is a set of lattice-valued of production; that is, \( P \) is a binary lattice-valued relationship on \( (P \cup T) \times (P \cup T) \times \Gamma \). For \( \alpha, \beta \in (V \cup T) \) and a fixed \( \theta \in L \), if \( P(\alpha, \beta) = \theta \) and \( \theta \neq 0 \), then it can be denoted by \( \alpha \rightarrow_\theta \beta \) which is called the production on \( \mathcal{G} \), and the production means that the membership degree that \( \alpha \) is defined as \( \beta \) is \( \theta \). \( \alpha \) is called the left part of \( \alpha \rightarrow_\theta \beta \), \( \beta \) is the right part of \( \alpha \rightarrow_\theta \beta \), and \( \theta \) is the membership degree that \( \alpha \) is defined as \( \beta \). The productions are termed the definitions or the grammar rules.

- \( S \in V \) is the start symbol of the grammar \( \mathcal{G} \).

According to Chomsky hierarchy of the grammar and the concept of the fuzzy grammar of Lee and Zadeh, the lattice-valued grammar can be divided into the following four types.

**Definition 2.** Let \( \mathcal{G} = (V, T, P, S) \) be a lattice-valued grammar. Then one has the following.

1. \( \mathcal{G} \) is called the lattice-valued type 0 grammar or the lattice-valued phrase structure grammar, if the production in \( P \) is without any constraint conditions.
The corresponding language $\varphi_\mathcal{G}$ is called the lattice-valued type 0 language or the lattice-valued phrase structure language.

(2) $\mathcal{G}$ is called the lattice-valued type 1 grammar or the lattice-valued context sensitive grammar, if, for any production $\alpha \rightarrow_\mathcal{G} \beta \in P$, one has $|\alpha| \leq |\beta|$. The corresponding language $\varphi_\mathcal{G}$ is called the lattice-valued type 1 language or the lattice-valued context sensitive language.

(3) $\mathcal{G}$ is called the lattice-valued type 2 grammar or the lattice-valued context-free grammar, if, for any production $\alpha \rightarrow_\mathcal{G} \beta \in P$, one has $|\alpha| \leq |\beta|$ and $\alpha \in V$. The corresponding language $\varphi_\mathcal{G}$ is called the lattice-valued type 2 language or the lattice-valued context-free language.

(4) $\mathcal{G}$ is called the lattice-valued type 3 grammars or the lattice-valued regular grammar, if, for any production $\alpha \rightarrow_\mathcal{G} \beta \in P$, one has

$$A \rightarrow_\mathcal{G} \omega, \quad A \rightarrow_\mathcal{G} \omega B \quad (\text{or } A \rightarrow_\mathcal{G} \omega \omega, A \rightarrow_\mathcal{G} B \omega),$$

where $A, B \in V, \omega \in T^*$. The corresponding language $\varphi_\mathcal{G}$ is called the lattice-valued type 3 language or the lattice-valued regular language.

**Definition 3.** Let $\mathcal{G} = (V, T, P, S)$ be a lattice-valued grammar. Define a binary lattice-valued relationship $\Rightarrow_\mathcal{G}$ on $(V \cup T)^*$ as follows:

$$\Rightarrow_\mathcal{G} \quad (\gamma \delta, \gamma \beta \delta) = P(\alpha, \beta),$$

for all $\gamma, \delta \in (V \cup T)^*$ and $\alpha \rightarrow_\mathcal{G} \beta \in P$, and it can be written as $\gamma \alpha \beta \delta = \gamma \beta \delta$.

If $\Rightarrow_\mathcal{G} (\gamma \alpha \delta, \gamma \beta \delta) = \theta \neq 0$, then we can say that the membership degree of $\gamma \alpha \delta \in \mathcal{G}$ can deduce $\gamma \beta \delta$ in the lattice-valued grammar $\mathcal{G}$ is $\theta$, or the membership degree of $\gamma \beta \delta$ can be reduced into $\gamma \alpha \delta$ in the lattice-valued grammar $\mathcal{G}$ is $\theta$.

$n_\mathcal{G}$ denotes the $n$ power of $\Rightarrow_\mathcal{G}$; that is, $n_\mathcal{G} \Rightarrow_\mathcal{G} = (\Rightarrow_\mathcal{G})^n$; $^*_\mathcal{G}$ denotes the positive closure of $\Rightarrow_\mathcal{G}$; that is, $^*_\mathcal{G} = (\Rightarrow_\mathcal{G})^*$. If $\Rightarrow_\mathcal{G} (\alpha, \beta) = \theta \neq 0$, then we can say that the membership degree of $\alpha \in \mathcal{G}$ after $n$ steps can deduce $\beta$ is $\theta$, or the membership degree of $\beta$ in $\mathcal{G}$ after $n$ steps can summarize $\alpha$ is $\theta$, it also can be written as $\alpha \Rightarrow_\mathcal{G} \beta$.

Let $\alpha, \beta \in (N \cup T)^*$; according to the composition definition of the binary lattice-valued relationship, it is not difficult to see the following.

If $\Rightarrow_\mathcal{G} (\alpha, \beta) = \theta \neq 0$, then we can say that the membership degree of $\alpha \in \mathcal{G}$ after $n$ steps can deduce $\beta$ is $\theta$, or the membership degree of $\beta$ in $\mathcal{G}$ after $n$ steps can summarize $\alpha$ is $\theta$, it also can be written as $\alpha \Rightarrow_\mathcal{G} \beta$.

If $\Rightarrow_\mathcal{G} (\alpha, \beta) = \theta \neq 0$, then we can say that the membership degree of $\alpha \in \mathcal{G}$ after at least 1 step can deduce $\beta$ is $\theta$, or the membership degree of $\beta$ in $\mathcal{G}$ after at least 1 step can summarize $\alpha$ is $\theta$, it also can be written as $\alpha \Rightarrow_\mathcal{G} \beta$.

**Definition 4.** Let $\mathcal{G} = (V, T, P, S)$ be a lattice-valued type 0 grammar, $\omega \in T^*$. Define

$$\varphi_\mathcal{G}(\omega) = (\Rightarrow_\mathcal{G})^*(S, \omega).$$

$\varphi_\mathcal{G}$ is called the lattice-valued language generated by lattice-valued type 0 grammar which is the lattice-valued type 0 language or lattice-valued phrase structure language. One calls that two lattice-valued grammars $\mathcal{G}_1, \mathcal{G}_2$ are equivalent, if they can generate the same lattice-valued languages; that is, $\varphi_{\mathcal{G}_1} = \varphi_{\mathcal{G}_2}$.

**5. The Equivalence between the Lattice-Valued Type 0 Grammar and the Lattice-Valued Turing Machine**

**Theorem 5.** If $\varphi$ can be recognized by the lattice-valued Turing machine $\mathcal{M} = (Q, \Gamma, \Sigma, \delta, q_0, B, F)$. that is, $\varphi = \varphi_\mathcal{G}$, then there exists the lattice-valued type 0 grammar $\mathcal{G}$ such that $\varphi = \varphi_\mathcal{G}$.

**Proof.** Suppose that $\varphi$ can be recognized by lattice-valued Turing machine $\mathcal{M} = (Q, \Gamma, \Sigma, \delta, q_0, B, F)$.

Formally, we construct a lattice-valued type 0 grammar $\mathcal{G} = (V, \Sigma, P, A_0)$, where

$$V = ((\Sigma \cup \epsilon) \times \Gamma) \cup \{A_1, A_2, A_3\} \cup Q$$

and the production in $P$ is as follows:

1. $A_1 \rightarrow q_0 A_2$, its membership degree is 1, and the following productions which are not labeled with membership degree were regarded as:
2. $A_2 \rightarrow \{a, a\} A_2, \forall a \in \Sigma$;
3. $A_2 \rightarrow A_3$;
4. $A_3 \rightarrow \{\epsilon, \epsilon\} A_2$;
5. $A_3 \rightarrow \epsilon$;
6. $q[a, X] \rightarrow q[a, Y] p, \forall a \in \Sigma \cup \{\epsilon\}, \forall q \in Q, \forall X, Y \in \Gamma$, and $\delta(q, X) = (p, \gamma, 1, \theta)$;
7. $[b, Z] q[a, X] \rightarrow q[a, Y] p, \forall a, b \in \Sigma \cup \{\epsilon\}, \forall q \in Q, \delta(q, X) = (p, \gamma, -1, \theta)$;
8. $[a, X] q \rightarrow q[a, X] q \rightarrow q[a, X] \rightarrow \epsilon, \forall a \in \Sigma \cup \{\epsilon\}, \forall X \in \Gamma, \forall q \in F$. 

By using rules (1) and (2), we can obtain
\[ A_1 \xrightarrow{\ast \varnothing} q_0 [a_1,\alpha_1] [a_2,\alpha_2] \cdots [a_n,\alpha_n] A_2, \]  
(27)
where every \( a_i \) belongs to \( \Sigma \). Assume that \( M \) can accept a string \( a_1, a_2, \ldots, a_n \), and there exists \( m \) such that \( M \) can use at most \( m \) units of the string input on the right. Using rules (3), (4) \((m \text{ times})\), and (5) in order, we can obtain
\[ A_1 \xrightarrow{\ast \varnothing} q_0 [a_1,\alpha_1] [a_2,\alpha_2] \cdots [a_n,\alpha_n] [\epsilon, B]^m. \]  
(28)
Then we only use rules (6) and (7), until an accepting state can be generated.

By the induction hypothesis, we can prove that, if
\[ q_0 a_1 a_2 \cdots a_n \xrightarrow{\varnothing} q X_1, X_2, \ldots, X_{r-1} q X_r, \ldots, X_t, \]  
(29)
then
\[ q_0 [a_1,\alpha_1] [a_2,\alpha_2] \cdots [a_n,\alpha_n] [\epsilon, B]^m \xrightarrow{\varnothing \theta} [a_1, X_1] [a_2, X_2] \cdots [a_{r-1}, X_{r-1}] q \xrightarrow{\varnothing \theta} [a_r, X_r] \cdots [a_{m+n}, X_{m+n}], \]  
(30)
for all \( a_1, a_2, \ldots, a_n \in \Sigma, a_{r-1} = a_{m+n+1} = \epsilon, X_1, X_2 \cdots X_{m+n} \in \Gamma, \) and \( X_{r-1} = X_{m+n+2} = X_{m+n} = B. \)

Obviously, for the 0 movement, the inductive hypothesis holds, since \( r = 1, s = n. \)

Assume that the inductive hypothesis holds for the \( k-1 \) movements. Let
\[ q_0 a_1 a_2 \cdots a_n \xrightarrow{\varnothing \theta_1} X_1, X_2, \ldots, X_{r-1} q X_r, \ldots, X_t \xrightarrow{\varnothing \theta_2} Y_1, Y_2, \ldots, Y_{r-1} q Y_r, \ldots, Y_s. \]  
(31)

By the induction hypothesis,
\[ q_0 [a_1,\alpha_1] [a_2,\alpha_2] \cdots [a_n,\alpha_n] [\epsilon, B]^m \xrightarrow{\varnothing \theta_1} [a_1, X_1] [a_2, X_2] \cdots [a_{r-1}, X_{r-1}] q \]  
(32)
\[ \xrightarrow{\varnothing \theta_2} [a_r, X_r] \cdots [a_{m+n}, X_{m+n}], \]
where each \( a_i \) and \( X_i \) satisfy the following conditions.

If \( t = r + 1 \), then the \( k \)th movement of \( M \) goes to the right, so
\[ \delta (q, X_r) = (p, Y_r + 1, \lambda), \]  
(33)
and the membership degree \( \theta_2 = \theta_1 \wedge \lambda. \) By rule (6),
\[ q [a_i,\alpha_i] \rightarrow_\lambda [a_r, Y_r] p \]  
(34)
is a production of \( \mathcal{G}. \) Therefore,
\[ q_0 [a_1,\alpha_1] [a_2,\alpha_2] \cdots [a_n,\alpha_n] [\epsilon, B]^m \xrightarrow{\varnothing \theta_2} [a_1, Y_1] [a_2, Y_2] \cdots [a_{r-1}, Y_{r-1}] p \quad \text{(35)} \]
\[ \times [a_r, Y_r] \cdots [a_{m+n}, Y_{m+n}], \]
for \( i > u, Y_i = B, \) and the membership degree \( \theta_2 = \theta_1 \wedge \lambda. \) If \( t = r + 1, \) then the \( k \)th movement of \( M \) goes to the left, we can apply rule (7) to prove the above equation, and we can obtain for the case of \( r - 1 \)
\[ \delta (q, X_r) = (p, Y_r - 1, \lambda). \]  
(36)

By rule (8), if \( p \) in \( F, \) then
\[ [a_1, Y_1] \cdots [a_{r-1}, Y_{r-1}] p [a_r, Y_r] \cdots [a_{m+n}, Y_{m+n}] \xrightarrow{\varnothing \theta} a_1 a_2 \cdots a_n. \]  
(37)
Thus
\[ \varphi_{\mathcal{G}}(\omega) = \xrightarrow{\varnothing \theta} (A_1, \omega) \]
\[ = \left\{ \varphi_{\mathcal{G}}[a_1, a_2, \ldots, a_n] X_1, X_2 \cdots X_{m+n} q X_r \cdots X_t \right\} \]
\[ = \left\{ \varphi_{\mathcal{G}}[a_1, a_2] \right\} \]
\[ = \left\{ \varphi_{\mathcal{G}}[a_1, a_2] \mid q \in F, a_1, a_2 \in \Gamma^* \right\} = \varphi_{\mathcal{M}}(\omega). \]  
(38)

Similarly, we have the following result.

**Theorem 6.** Assume \( \varphi \) is a language \( \varphi_{\mathcal{G}} \) which generated by the lattice-valued type 0 grammar \( \mathcal{G} = (V, T, P, S) \), and then, \( \varphi \) can be recognized by the lattice-valued Turing machine \( M = (Q, \Gamma, \Sigma, \delta, q_0, B, F) \); that is, \( \varphi = \varphi_{\mathcal{M}}. \)

**Conflict of Interests**

The authors declare that there is no conflict of interests regarding the publication of this paper.

**Acknowledgments**

The authors thank reviewers for their patient and invaluable advice, all of which have been of inestimable worth to the completion of their paper. They also thank editor for being serious and responsible. The authors acknowledge the support of the National Natural Science Foundation of China (nos. 1161050, 31240020).
References


Submit your manuscripts at http://www.hindawi.com