An Improved Proof of the Closure Under Homomorphic Inverse of FCFL Valued in Lattice-Ordered Monoids

Haihui Wang, Luyao Zhao, and Ping Li(✉)

College of Mathematics and Information Science, Shaanxi Normal University, Xi'an 710062, China
{wanghaihui,zhaoluyao,liping}@snnu.edu.cn

Abstract. We study fuzzy context-free grammars (FCFG), fuzzy context-free languages (FCFL) and fuzzy pushdown automata (FPDA) valued in lattice-ordered monoids. Inspired by the ideas of crisp cases, we get similar results, but some of them depend on commutative law. Particularly, we give two proofs of the closure under homomorphic inverse of FCFL when lattice-ordered monoids are commutative. Comparing with the classical method, we show that the improved proof is more efficient.

Keywords: Lattice-ordered monoid · Fuzzy context-free language · Fuzzy pushdown automata · Homomorphic inverse · Closure property

1 Introduction

The characterization of formal languages is a very important aspect of research in the classical computation theory. For example, regular languages can be characterized by some finite automata, regular expressions and regular grammars. However, the problem of vagueness and imprecision are frequently encountered in the study of natural languages. In order to represent the imprecision of natural languages, various fuzzy languages have been proposed. By introducing the concept of fuzziness into the structure of formal grammars, the concept of fuzzy automata was introduced by Santos [22,23] as early as in the late 1960s, and after the concept of fuzzy grammar was introduced by Lee and Zadeh [13,14]. There are amount of work on fuzzy languages, such as the relationships between fuzzy grammars and fuzzy languages, the relations of fuzzy automata and fuzzy language, and the algebraic properties of fuzzy languages [2,17,18]. As a further extension, Kim et al. put forward one type of L-fuzzy grammar based on the distributive lattice and Boolean lattice, another type of L-fuzzy grammar based on lattice-ordered monoid by assigning the element of lattice to the rewriting rules of a formal grammar [11]. Gerla also studied fuzzy grammars and recursively enumerable fuzzy languages, and proved that a fuzzy language can be

Supported by the National Natural Science Foundation of China under Grant 11301321, Grant 61673250, Grant 61672023.

X. Sun et al. (Eds.): NCTCS 2019, CCIS 1069, pp. 64–75, 2019.
https://doi.org/10.1007/978-981-15-0105-0_5

generated by a fuzzy grammar if and only if it is recursively enumerable [6]. As one of the generators of fuzzy languages, fuzzy grammars have been used to solve some important questions such as intelligent interface design (Senay) [25], lexical analysis, clinical monitoring (Steimann and Adlassning) [26], neural networks (Giles et al.) [19], and pattern recognition (Depalma and Yau) [4]. It is known that the context-free grammars are powerful than regular grammars, also the classical pushdown automata and finite automata [9,24]. Therefore, fuzzy pushdown automata and fuzzy context-free grammars were discussed in [3,12,16,24,27,28] and automata theory based on residuated lattice-valued logic has been investigated recently [21,28]. It is shown that their many properties are similar to classical pushdown automata, context-free grammars and context-free languages. In particular, we can get a much more efficient way to prove the closure under homomorphic inverse of fuzzy context-free languages.

In this paper, fuzzy context-free languages whose codomain forms a commutative lattice-ordered monoid $\mathcal{L}$ are studied. Furthermore, we show that the homomorphic inverse of fuzzy context-free languages are also fuzzy context-free languages by two ways, and the latter is much more efficient than the former by comparison.

The rest of the paper is arranged as follows. In Sect. 2, we introduce lattice-ordered monoids and give some examples. In Sect. 3, we discuss fuzzy context-free grammars, fuzzy context-free languages and their relationships. Section 4 studies the fuzzy pushdown automata valued in a lattice-ordered monoid $\mathcal{L}$. In Sect. 5, comparing with the classical cases, we give an improved proof of the closure under homomorphic inverse of fuzzy context-free languages, which increases the efficiency of operation. Finally, some conclusions are concerned in Sect. 6.

2 Lattice-Ordered Monoids

We first introduce the definition of lattice-ordered monoid and give some examples.

Definition 2.1. *Given a lattice $\mathcal{L}$, we use $\vee, \wedge$ to represent the supremum operation and infimum operation on $\mathcal{L}$, respectively. We need $\mathcal{L}$ to have the least and largest elements in the paper, which will be denoted as 0, 1, respectively. Assume that there is a binary operation $\bullet$ (we call it multiplication) on $\mathcal{L}$ such that $(\mathcal{L}, \bullet, e)$ is a monoid with identity $e \in \mathcal{L}$.*

We call $\mathcal{L}$ an po-monoid (some modification of the notion of partially ordered monoid in [5]) if it satisfies the following two conditions for any $a, b, x \in \mathcal{L}$,

(1) $\forall a \in \mathcal{L}, a \bullet 0 = 0 \bullet a = 0$,
(2) $a \leq b \Rightarrow a \bullet x \leq b \bullet x$ and $x \bullet a \leq x \bullet b$.

And we call $\mathcal{L}$ a lattice-ordered monoid or l-monoid if $\mathcal{L}$ is a po-monoid and it satisfies the distributive laws, i.e.,

(3) $\forall a, b, c \in \mathcal{L}, a \bullet (b \vee c) = (a \bullet b) \vee (a \bullet c)$ and $(b \vee c) \bullet a = (b \bullet a) \vee (c \bullet a)$.

Moreover, if $\mathcal{L}$ is a complete lattice, and it satisfies the following infinite distributive laws,

(4) $a \bullet (\bigvee_t b_t) = \bigvee_t (a \bullet b_t)$ and $(\bigvee_t b_t) \bullet a = \bigvee_t (b_t \bullet a)$,

where T is an index set, then we call $\mathcal{L}$ a quantale. If the distributive laws in (3) holds only for countable set $\{b_t\}_{t\in T}$ then $\mathcal{L}$ is called a countable lattice-ordered monoid.

We call the lattice-ordered monoid $(\mathcal{L}, \bullet, \vee)$ is commutative, if $a \bullet b = b \bullet a$ holds for any $a, b \in \mathcal{L}$. For a lattice-ordered monoid, we only concern the multiplication $\bullet$ and finite supremum operation $\vee$, in what follows, a lattice-ordered is denoted as $(\mathcal{L}, \bullet, \vee)$. If we deal with the subalgebra $\mathcal{L}_1$ of a lattice-ordered monoid$(\mathcal{L}, \bullet, \vee)$, it means that $\mathcal{L}_1$ is a nonempty subset of $\mathcal{L}$ and $\mathcal{L}_1$ is closed under the multiplication and finite supremum of $\mathcal{L}$. We do not concern with the infimum operation in $\mathcal{L}_1$.

The followings are discussed on the lattice-ordered monoid $\mathcal{L}$ without special instructions.

Example 2.1.

(1) Let $(\mathcal{L}, \wedge, \vee)$ be a distributive lattice, and let $\wedge = \bullet$, then $\mathcal{L}$ is a lattice-ordered monoid, and the identity of multiplication is 1.

(2) Let $(\mathcal{L}, \bullet, \vee)$ be a lattice-ordered monoid, the identity is e. We use $L(n)$ to denote all $n \times n$ matrices with values in $\mathcal{L}$. The multiplication, denote as $\circ$, is defined as Sup$-\bullet$ composition; and $\vee$ is the pointwise-$\vee$. That is, for two $n \times n$ matrices, $A = (a_{ij}), B = (b_{ij})$, with values in $\mathcal{L}$, let $A \circ B = C = (c_{ij})$, then $c_{ij} = \vee_{k=1}^{n}(a_{ik} \bullet b_{kj})$ and let $A \vee B = D = (d_{ij}), d_{ij} = a_{ij} \vee b_{ij}$. Then $(L(n), \circ, \vee)$ is also a lattice-ordered monoid, the identity is the diagonal-matrix $E = diag(e, \cdots, e)$, with e as the diagonal element. In general, the multiplication on $L(n)$ is not commutative, even if the multiplication on $\mathcal{L}$ is commutative.

(3) Let $\bullet$ be any uninorm on [0,1]. If $0 \bullet 1 = 0$, then$([0, 1], \bullet, \vee)$ is a commutative lattice-ordered monoid. In particular, if $\bullet$ is a t-norm on [0,1], then $0 \bullet 1 = 0$, then $([0, 1], \bullet, \vee)$ is a commutative lattice-ordered monoid with identity $e = 1$.

(4) Complete residuated lattices are special kinds of quantales, where its multiplication is commutative and the identity is the same as the largest element.

3 Fuzzy Grammars

In this section, we study fuzzy context-free grammars, fuzzy context-free languages and their relationships.

Definition 3.1. *Fuzzy grammar is a four tuple $G = (N, T, P, S)$, where N is a finite nonterminal alphabet and its elements are called variables; T is a finite terminal alphabet, $N \cap T = \emptyset$; $S \in N$ is a start symbol; P is a finite alphabet of fuzzy productions $u \to^{\rho} v$, where $u \in (N \bigcup T)^* N (N \bigcup T)^*$, $v \in (N \bigcup T)^*$, and $\rho \in \mathcal{L} - \{0\}$ represents the membership value of rewriting rule $u \to v$. We suppose S only appears in the left of fuzzy production $u \to v$.*

For $u_1, \cdots, u_n \in (N \bigcup T)^*$, if $u_1 \Rightarrow^{\rho_1} u_2 \Rightarrow^{\rho_2} \cdots \Rightarrow^{\rho_{n-1}} u_n$, then u_n called the derivation chain from u_1, denoted as $u_1 \Rightarrow_*^{\rho} u_n$, where $\rho = \rho_1 \bullet \rho_2 \bullet \cdots \bullet \rho_{n-1}$.

Definition 3.2. *The fuzzy language $L(G) : T^* \to \mathcal{L}$ generated by fuzzy grammar G is defined as, for any $\theta \in T^*$,*

$$L(G)(\theta) = \vee\{\rho | S \Rightarrow_*^{\rho} \theta\}.$$

Fuzzy grammar, also called type 0 grammar or fuzzy phrase grammar, it is further classified as follows.

Suppose that $G = (N, T, P, S)$ is a fuzzy grammar, then

(1) G is called fuzzy context-sensitive grammar (FCSG) or type 1 grammar, if for any $u \to^{\rho} v \in P$, there is $|u| \leq |v|$. And $L(G)$ is called fuzzy context-sensitive language (FCSL).
(2) G is called fuzzy context-free grammar (FCFG) or type 2 grammar, if for any $u \to^{\rho} v \in P$, there is $|u| \leq |v|$ and $u \in N$. And $L(G)$ is called fuzzy context-free language (FCFL).
(3) G is called fuzzy regular grammar (FRG) or type 3 grammar, if for any $u \to^{\rho} v \in P$, there is $u \in N$ and $v \in TB$, $B \in N \cup \{\varepsilon\}$, or $u = S, v = \varepsilon$. And $L(G)$ is called fuzzy context-free language (FRL).

In the paper, we only concern with the fuzzy context-free grammar.

Definition 3.3. *Suppose that $G = (N, T, P, S)$ is a fuzzy context-free grammar, then G is called fuzzy Chomsky normal form (FCNF) if for any production formula of G they have the form:*

$$A \to^{\rho} BC \text{ or } A \to^{\rho} a \text{ or } S \to^{\rho} \varepsilon,$$

where $A, B, C \in N, a \in T, \rho \in \mathcal{L} - \{0\}$.

Theorem 4.1. *For any fuzzy context-free grammar, there is an equivalent Chomsky normal form.*

4 Fuzzy Pushdown Automata

Now, we give the fuzzy pushdown automata valued in a lattice-ordered monoid $\mathcal{L}$.

Definition 4.1. *Fuzzy pushdown automata (FPDA) is a seven tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, where Q, Σ, Γ are nonempty finite sets, and they represent state sets, input alphabet, stack alphabet, respectively. $q_0 \in Q, Z_0 \in \Gamma$ represent initial state and start symbol. $F : Q \to \mathcal{L}$ is a fuzzy final state, $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to F(Q \times \Gamma^*)$ which image is a finite fuzzy subset of $Q \times \Gamma^*$ is called fuzzy transition function.*

For any $p, q \in Q, \sigma \in \Sigma, Z \in \Gamma, \gamma \in \Gamma^*, \delta(q, \sigma, Z)(p, \gamma)$ means the possible degree that automata can enter next state p and the top stack letter Z transfer to γ when current state is q, the top stack letter is Z and the input symbol is σ. Similarly, $\delta(q, \varepsilon, Z)(p, \gamma)$ means the possible degree that the automata turn to the next state p and the top stack letter Z transfer to γ when current state is q, the top stack letter is Z and the input symbol is empty string. What's more, for any $(q, w, \gamma) \in Q \times \Sigma^* \times \Gamma^*$, it is called a instantaneous description of M and ID for short, it means that M is on the current state q, w is the untreated input string and M is focusing on the first character of w, the string in the stack is γ.

For $\alpha, \beta \in \Gamma^*$, β is the tail of α if there exists $\gamma \in \Gamma^*$ such that $\alpha = \gamma\beta$, denoted $\beta \leq \alpha$ and $\gamma = \alpha \setminus \beta$, $head(\beta)$ is the first character of β. Then we give the extension ∇ of δ as follows.

Definition 4.2. *Given a fuzzy pushdown automata $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, we define the fuzzy relation ∇ in $Q \times \Sigma^* \times \Gamma^*$ as,*

$$\nabla((p, w, \beta), (q, v, \alpha)) = \begin{cases} \delta(p, \varepsilon, head(\beta))(q, \alpha \setminus tail(\beta)), & v = w, tail(\beta) \leq \alpha, \\ \delta(p, head(w), head(\beta)(q, \alpha \setminus tail(\beta)), & v = tail(w), tail(\beta) \leq \alpha, \\ 0, & otherwise. \end{cases}$$

∇^* is defined as the reflexive and transitive closure of ∇. The reflexive and transitive closure of fuzzy relation R in set Q is defined as $R^* = I \cup R \cup R \circ R \cup \cdots \cup R^n \cup \cdots$, where $\circ$ is $Sup - \bullet$ composition of fuzzy relations.

Definition 4.3. *Given a fuzzy pushdown automata $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, for all $\theta \in \Sigma^*$,*
(1) *The fuzzy language accepted by M by final state is defined as,*

$$L(M)(\theta) = \bigvee_{q \in Q, \gamma \in \Gamma^*} [\nabla^*((q_0, \theta, Z_0), (q, \varepsilon, \gamma)) \bullet F(q)].$$

(2) *The fuzzy language accepted by M by empty stack is defined as,*

$$N(M)(\theta) = \bigvee_{q \in Q} \nabla^*((q_0, \theta, Z_0), (q, \varepsilon, \varepsilon)).$$

Theorem 4.1. *For an FPDA that accepts a fuzzy language L by empty stack, there exists another FPDA that accepts L by final states. And the vise versa.*

Corollary 4.1. *The fuzzy final states can be taken as crisp states when the FPDA accept fuzzy language by the final state.*

Theorem 4.2. *For a fuzzy language, it can be accepted by an FPDA iff the fuzzy language is FCFL.*

5 The Properties of Fuzzy Context-Free Language

this section, we investigate some properties of fuzzy context-free languages, and except the closure under homomorphic inverse, the proofs of other conclusions are similar to the classical conditions, so we don't give their proofs in this paper.

Definition 5.1. *Let f, g be any fuzzy context-free language on Σ^* that valued in a lattice-ordered monoid $\mathcal{L}$, and $a \in \mathcal{L}, \theta \in \Sigma^*$, then we defined the operations as follows:*

(1) The scalar product of a and f, denoted af and fa, is defined as, $(af)(\theta) = a \bullet f(\theta)$ and $(fa)(\theta) = f(\theta) \bullet a$.
(2) The union of f_1 and f_2, denoted $f_1 \cup f_2$, is defined as, $(f_1 \cup f_2)(\theta) = f_1(\theta) \vee f_2(\theta)$.
(3) The concatenation of f_1 and f_2, denoted $f_1 f_2$, is defined as, $(f_1 f_2)(\theta) = \bigvee_{\theta_1\theta_2=\theta}[f_1(\theta_1) \bullet f_2(\theta_2)]$. And then the concatenation operation satisfies the associative laws.

Definition 5.2. *Let Σ, Δ be finite nonempty character sets, $\phi : \Sigma \to F(\Delta^*)$ is a mapping, and ϕ is called fuzzy context-free substitution if $\phi(\sigma)$ is a fuzzy context-free language of Δ for any $\sigma \in \Sigma$.*

Given a mapping $\phi : \Sigma \to F(\Delta^*)$, then ϕ can be extended on Σ^* by the way as follows,

(1) $\phi(\varepsilon) = \{\frac{1}{\varepsilon}\}$;
(2) $\forall\theta \in \Sigma^*, \forall\sigma \in \Sigma, \phi(\theta\sigma) = \phi(\theta)\phi(\sigma)$.

Then we extend ϕ on $F(\Sigma^*)$, denoted as, $\phi : F(\Sigma^*) \to F(\Delta^*)$, and $\forall h \in F(\Sigma^*), \forall\omega \in \Delta^*$,

$$\phi(h)(\omega) = \bigvee_{\theta\in\Sigma^*} [h(\theta) \bullet \phi(\theta)(\omega)].$$

Definition 5.3. *A mapping $\phi : \Sigma \to F(\Delta^*)$ is called fuzzy homomorphism, if for any $\sigma \in \Sigma$, there exists unique $w \in \Sigma^*, a \in \mathcal{L} - \{0\}$ such that $\phi(\sigma) = \frac{a}{w}$. For the above mapping ϕ and $\sigma \in \Sigma$, we can define two mappings $\phi_1 : \Sigma \to \Delta^*$ and $\phi_2 : \Sigma \to \mathcal{L}$ as, $\phi_1(\sigma) = w$, $\phi_2(\sigma) = a$, respectively. Then ϕ_1 is the usual homomorphism, therefore, $\forall\sigma \in \Sigma, \phi(\sigma) = \frac{\phi_2(\sigma)}{\phi_1(\sigma)}$.*

Then we can extend ϕ_1 and ϕ_2 on Σ^* as, $\forall\theta = \sigma_1 \cdots \sigma_k \in \Sigma^*$, $\phi_1(\sigma_1 \cdots \sigma_k) = \phi_1(\sigma_1) \cdots \phi_1(\sigma_k)$, and $\phi_2(\sigma_1 \cdots \sigma_k) = \phi_2(\sigma_1) \bullet \cdots \bullet \phi_2(\sigma_k)$. Besides, we declare that $\phi_1(\varepsilon) = \varepsilon$, $\phi_2(\varepsilon) = e$.

Thus, we can extend ϕ on $F(\Sigma^*)$ as,

$$\forall g \in F(\Sigma^*), \quad \forall\omega \in \Delta^*, \quad \phi(g)(\omega) = \vee\{g(\theta) \bullet \phi_2(\theta) | \phi_1(\theta) = \omega\},$$

at the same time, we define the inverse mapping $\phi^{-1} : F(\Delta^*) \to F(\Sigma^*)$ as,

$$\forall h \in F(\Delta^*), \quad \forall\theta \in \Sigma^*, \quad \phi^{-1}(h)(\theta) = h(\phi_1(\theta)) \bullet \phi_2(\theta).$$

Theorem 5.1. *Any FCFL is still an FCFL under the fuzzy context-free substitution.*

By the extensive definition of fuzzy homomorphism, we know that fuzzy context-free homomorphism is a special type of fuzzy context-free substitution, so we can get the following conclusion.

Corollary 5.1. *Suppose that* $\phi : \Sigma \to F(\Delta^*)$ *is a fuzzy homomorphism, then* $\phi(f)$ *is an FCFL on* Δ *if* f *is an FCFL on* Σ.

That is to say, fuzzy context-free languages are closed under the fuzzy homomorphism, now we consider the fuzzy homomorphic inverse of fuzzy context-free languages in Definition 5.2. Suppose that f is a fuzzy context-free language on Δ, $\phi : \Sigma \to F(\Delta^*)$ is a fuzzy homomorphism, then f can be described by a fuzzy context-free grammar and accepted by a fuzzy pushdown automata. If $\phi^{-1}(f)$ is a fuzzy context-free language, there must be a fuzzy context-free grammar and a fuzzy pushdown automata corresponding to it. Here we only consider designing a fuzzy pushdown automata to accept it. Assume that M_2 is a fuzzy pushdown automata that accepts f. We need construct a fuzzy pushdown automata M_1 which can simulate the processing of M_2 to $\phi_1(a)$ when M_1 reads character a. But, we can't ignore the degree of the processing, that is $\phi_2(a)$. For one thing, we need to store $\phi_1(a)$ in the finite controller of M_1, and the degree of this process is $\phi_2(a)$. For another thing, $\phi_1(a)$ is a string, we need to simulate the process of M_2 to $\phi_1(a)$. Now we give two ways to complete the process, and the second solution is an improved way.

(1) We can use an empty move of M_1 to simulate the process of M_2 to each character of $\phi_1(a)$, and M_2 running $|\phi_1(a)|$ steps when it finished processing $\phi_1(a)$. This way is similar to the process of classical case, but what we should concern about is just the possible degree each transition of M_2.
(2) We can use an empty move of M_1 to simulate the process of M_2 to the whole string $\phi_1(a)$, then let M_1 remember the state and stack letter, which are M turns to after it finished processing $\phi_1(a)$. In the process, the possible transition degree of M_1 is just the possible degree that M_2 processed string $\phi_1(a)$.

Remark 5.1. *From the definition of* ∇^*, *it is clearly that if* $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ *is a fuzzy pushdown automata,* $p, q \in Q, x, y \in \Sigma^*, \alpha, \beta \in \Gamma^*$, *then* $\nabla^*((q, x, \alpha), (p, y, \beta)) = \nabla^*((q, x\omega, \alpha), (p, y\omega, \beta))$ *holds for any* $\omega \in \Sigma^*$.

We will frequently use the fact to prove the following theorem.

Theorem 5.2. *If* $\mathcal{L}$ *is commutative, then the homomorphic inverse of FCFL is FCFL.*

Proof I. Let f be a fuzzy context-free language of Δ, $\phi : \Sigma \to F(\Delta^*)$ is a fuzzy homomorphism. Then there is a fuzzy pushdown automata $M_2 = (Q_2, \Delta, \Gamma, \delta_2, q_0, Z_0, \{q_f\})$ which accepts f, that is, for any θ in Δ^*, $f(\theta) = L(M_2)(\theta) = \bigvee_{\gamma \in \Gamma^*} \nabla_2^* ((q_0, \theta, Z_0), (q_f, \varepsilon, \gamma))$. Now we construct a fuzzy pushdown automata $M_1 = (Q_1, \Delta, \Gamma, \delta_1, [q_0, \varepsilon], Z_0, [q_f, \varepsilon])$ as,

(1) $Q_1 = \{[q,x] | \forall a \in \Sigma \bigcup \{\varepsilon\}, x \text{ is a suffix of } \phi_1(a)\}$, where the number of the suffix of $\phi_1(a)$ is finite, and Q_2 is a finite set, so Q_1 is finite.
(2) $\forall q \in Q_2, a \in \Sigma \bigcup \{\varepsilon\}, Z \in \Gamma, \delta_1([q,\varepsilon],a,Z)([q,\phi_1(a)],Z) = \phi_2(a)$.
(3) $\forall q,p \in Q_2, u \in \Sigma \cup \{\varepsilon\}, \gamma \in \Gamma^*, \delta_1([q,ux],\varepsilon,Z)([p,x],\gamma) = \delta_2(q,u,Z)(p,\gamma)$, and for other cases, $\delta_1 = 0$.

Therefore, if $\theta = \varepsilon$, then

$$\begin{aligned}
L(M_1)(\varepsilon) &= \bigvee_{\gamma \in \Gamma^*} \nabla_1^*(([q_0,\varepsilon],\varepsilon,Z_0),([q_f,\varepsilon],\varepsilon,\gamma)) \\
&= \bigvee_{\gamma \in \Gamma^*} \nabla_1(([q_0,\varepsilon],\varepsilon,Z_0),([q_0,\phi_1(\varepsilon)],\varepsilon,Z_0)) \bullet \nabla_1(([q_0,\phi_1(\varepsilon)],\varepsilon,Z_0),([q_f,\varepsilon],\varepsilon,\gamma)) \\
&= \bigvee_{\gamma \in \Gamma^*} \delta_1([q_0,\varepsilon],\varepsilon,Z_0)([q_0,\phi_1(\varepsilon)],Z_0) \bullet \delta_1([q_0,\phi_1(\varepsilon)],\varepsilon,Z_0)([q_f,\varepsilon],\gamma) \\
&= \bigvee_{\gamma \in \Gamma^*} \nabla_2^*((q_0,\phi_1(\varepsilon),Z_0),(q_f,\varepsilon,\gamma)) \bullet \phi_2(\varepsilon) \\
&= f(\phi_1(\varepsilon)) \bullet \phi_2(\varepsilon) \\
&= \phi^{-1}(f)(\varepsilon).
\end{aligned}$$

Note that for any $\theta \in \Sigma^+$, let $\theta = a_1 \cdots a_n$, $a_i \in \Sigma$, $\phi_1(a_i) = b_{i1} \cdots b_{im}$, $i = 1,2,...,n, |\phi_1(a_i)| = m$, then $\phi_1(\theta) = \phi_1(a_1) \cdots \phi_1(a_n)$, $\phi_2(\theta) = \phi_2(a_1) \bullet \cdots \bullet \phi_2(a_n)$, therefore,

$$\begin{aligned}
L(M_1)(\theta) &= \bigvee_{\gamma \in \Gamma^*} \nabla_1^*(([q_0,\varepsilon],\theta,Z_0),([q_f,\varepsilon],\varepsilon,\gamma)) \\
&= \bigvee_{\gamma_i \in \Gamma^*, q_i \in Q_2} [\nabla_1(([q_0,\varepsilon],a_1 \cdots a_n,Z_0),([q_0,\phi_1(a_1)],a_2 \cdots a_n,Z_0)) \\
&\bullet \nabla_1(([q_0,\phi_1(a_1)],\varepsilon a_2 \cdots a_n,Z_0),([q_1,\varepsilon],a_2 \cdots a_n,\gamma_1))] \\
&\bullet [\nabla_1(([q_1,\varepsilon],a_2 \cdots a_n,\gamma_1),([q_1,\phi_1(a_2)],a_3 \cdots a_n,\gamma_1)) \\
&\bullet \nabla_1(([q_1,\phi_1(a_2)],\varepsilon a_3 \cdots a_n,\gamma_1),([q_2,\varepsilon],a_3 \cdots a_n,\gamma_2))] \bullet \cdots
\end{aligned}$$

$$\begin{aligned}
&\bullet [\nabla_1(([q_{n-1},\varepsilon],a_n,\gamma_{n-1}),([q_{n-1},\phi_1(a_n)],\varepsilon,\gamma_{n-1})) \\
&\bullet \nabla_1(([q_{n-1},\phi_1(a_n)],\varepsilon,\gamma_{n-1}),([q_f,\varepsilon],\varepsilon,\gamma))] \\
&= \bigvee_{\gamma,\gamma_i \in \Gamma^*, q_i \in Q_2} [\delta_1(([q_0,\varepsilon],a_1,Z_0)([q_0,\phi_1(a_1)],Z_0) \\
&\bullet \delta_1([q_0,b_{11} \cdots b_{1m}],\varepsilon,Z_0)([q_1,\varepsilon],\gamma_1)] \bullet \cdots \\
&\bullet [\delta_1([q_{n-1},\varepsilon],a_n,\gamma_{n-1})([q_{n-1},\phi_1(a_n)],\gamma_{n-1}) \\
&\bullet \delta_1([q_{n-1},b_{n1} \cdots b_{nm}],\varepsilon,\gamma_{n-1})([q_f,\varepsilon],\gamma)] \\
&= \bigvee_{\gamma,\gamma_i \in \Gamma^*, q_i, q_{ij} \in Q_2} [\phi_2(a_1) \bullet \delta_1([q_0,b_{11} \cdots b_{1m}],\varepsilon,Z_0)([q_{01},b_{12} \cdots b_{1m}],\gamma_{01}) \\
&\bullet \delta_1([q_{01},b_{12} \cdots b_{1m}],\varepsilon,\gamma_{01})([q_{02},b_{13} \cdots b_{1m}],\gamma_{02}) \bullet \cdots \\
&\bullet \delta_1([q_{0(m-1)},b_{1m}],\varepsilon,\gamma_{0(m-1)})([q_1,\varepsilon],\gamma_1)]
\end{aligned}$$

$$
\begin{aligned}
&\bullet \cdots \bullet \\
&\phi_2(a_n) \bullet [\delta_1([q_{n-1}, b_{n1} \cdots b_{nm}], \varepsilon, \gamma_{(n-1)})([q_{(n-1),1}, b_{n2} \cdots b_{nm}], \gamma_{(n-1),1}) \\
&\bullet \delta_1([q_{(n-1),1}, b_{n2} \cdots b_{nm}], \varepsilon, \gamma_{(n-1),1})([q_{(n-1),2}, b_{n3} \cdots b_{nm}], \gamma_{(n-1),2}) \bullet \cdots \\
&\bullet \delta_1([q_{(n-1)(m-1)}, b_{nm}], \varepsilon, \gamma_{(n-1)(m-1)})([q_f, \varepsilon], \gamma)] \\
&= \bigvee_{\gamma, \gamma_i \in \Gamma^*, q_i, q_{ij} \in Q_2} [\delta_2(q_0, b_{11}, Z_0)(q_{01}, \gamma_{01}) \bullet \delta_2(q_{01}, b_{12}, \gamma_{01})(q_{02}, \gamma_{02}) \bullet \cdots \\
&\bullet \delta_2(q_{0,(m-1)}, b_{1m}, \gamma_{0,(m-1)})(q_1, \gamma_1)] \\
&\bullet \cdots \bullet \\
&[\delta_2(q_{n-1}, b_{n1}, \gamma_{n-1})(q_{(n-1),1}, \gamma_{(n-1),1}) \bullet \delta_2(q_{(n-1),1}, b_{n2}, \gamma_{(n-1),1})(q_{(n-1),2}, \gamma_{(n-1),2}) \bullet \cdots \\
&\bullet \delta_2(q_{(n-1)(m-1)}, b_{nm}, \gamma_{(n-1)(m-1)})(q_f, \gamma)] \\
&\bullet [\phi_2(a_1) \bullet \cdots \bullet \phi_2(a_n)] \\
&= \bigvee_{\gamma, \gamma_i \in \Gamma^*, q_i \in Q_2} [\nabla_2^*((q_0, b_{11} \cdots b_{1m}, Z_0), (q_1, \varepsilon, \gamma_1))] \\
&\bullet [\nabla_2^*((q_1, b_{21} \cdots b_{2m}, \gamma_{10}), (q_2, \varepsilon, \gamma_2))] \bullet \cdots \\
&\bullet [\nabla_2^*((q_{n-1}, b_{n1} \cdots b_{nm}, \gamma_{n-1}), (q_f, \varepsilon, \gamma))] \\
&\bullet [\phi_2(a_1) \bullet \cdots \bullet \phi_2(a_n)] \\
&= \bigvee_{\gamma, \gamma_i \in \Gamma^*, q_i \in Q_2} [\nabla_2^*((q_0, \phi_1(a_1), Z_0), (q_1, \varepsilon, \gamma_1)) \\
&\bullet \nabla_2^*((q_1, \phi_1(a_2), \gamma_1), (q_2, \varepsilon, \gamma_2)) \bullet \cdots \\
&\bullet \nabla_2^*((q_{n-1}, \phi_1(a_n), \gamma_{n-1}), (q_f, \varepsilon, \gamma))] \\
&\bullet [\phi_2(a_1) \bullet \cdots \bullet \phi_2(a_n)] \\
&= \bigvee_{\gamma, \gamma_i \in \Gamma^*, q_i \in Q_2} [\nabla_2^*((q_0, \phi_1(a_1)\phi_1(a_2) \cdots \phi_1(a_n), Z_0), (q_1, \varepsilon\phi_1(a_2) \cdots \phi_1(a_n), \gamma_1)) \\
&\bullet \nabla_2^*((q_1, \phi_1(a_2)\phi_1(a_3) \cdots \phi_1(a_n), \gamma_1), (q_2, \varepsilon\phi_1(a_3) \cdots \phi_1(a_n), \gamma_2)) \bullet \cdots \\
&\bullet \nabla_2^*((q_{n-1}, \phi_1(a_n), \gamma_{n-1}), (q_f, \varepsilon, \gamma))] \bullet \phi_2(\theta) \\
&= \bigvee_{\gamma \in \Gamma^*} [\nabla_2^*((q_0, \phi_1(a_1) \cdots \phi_1(a_n), Z_0), (q_f, \varepsilon, \gamma))] \bullet \phi_2(\theta) \\
&= f(\phi_1(\theta)) \bullet \phi_2(\theta) \\
&= \phi^{-1}(f)(\theta).
\end{aligned}
$$

Thus, $L(M_1) = \phi^{-1}(f)$.

Proof II. Let f be a fuzzy context-free language of Δ, $\phi : \Sigma \to F(\Delta^*)$ is a fuzzy homomorphism. Then there is a fuzzy pushdown automata $M_2 = (Q_2, \Delta, \Gamma, \delta_2, q_0, Z_0, \{q_f\})$ which accepts f. Now we construct a fuzzy pushdown automata $M_1 = (Q_1, \Delta, \Gamma, \delta_1, [q_0, \varepsilon], Z_0, [q_f, \varepsilon])$ as,

(1) $Q_1 = \{[q, x] | \forall a \in \Sigma \bigcup \{\varepsilon\}, x = \phi_1(a)\}$, where the number of $\phi_1(a)$ is finite, and Q_2 is a finite set, so Q_1 is finite.
(2) $\forall q \in Q_2, a \in \Sigma \bigcup \{\varepsilon\}, Z \in \Gamma, \delta_1([q, \varepsilon], a, Z)([q, \phi_1(a)], Z) = \phi_2(a)$,
(3) $\forall q, p \in Q_2, \gamma \in \Gamma^*, \delta_1([q, \phi_1(a)], \varepsilon, Z)([p, \varepsilon], \gamma) = \nabla_2^*((q, \phi_1(a), Z), (p, \varepsilon, \gamma))$, and for other cases, $\delta_1 = 0$.

Now we prove that $L(M_1) = \phi^{-1}(f)$. First, if $\theta = \varepsilon$, then

$$\begin{aligned}
L(M_1)(\varepsilon) &= \bigvee_{\gamma \in \Gamma^*} \nabla_1^*(([q_0, \varepsilon], \varepsilon, Z_0), ([q_f, \varepsilon], \varepsilon, \gamma)) \\
&= \bigvee_{\gamma \in \Gamma^*} \nabla_1(([q_0, \varepsilon], \varepsilon, Z_0), ([q_0, \phi_1(\varepsilon)], \varepsilon, Z_0)) \bullet \nabla_1(([q_0, \phi_1(\varepsilon)], \varepsilon, Z_0), ([q_f, \varepsilon], \varepsilon, \gamma)) \\
&= \bigvee_{\gamma \in \Gamma^*} \nabla_2^*((q_0, \phi_1(\varepsilon), Z_0), (q_f, \varepsilon, \gamma)) \bullet \phi_2(\varepsilon) \\
&= f(\phi_1(\varepsilon)) \bullet \phi_2(\varepsilon) \\
&= \phi^{-1}(f)(\varepsilon).
\end{aligned}$$

Note that for any $\theta \in \Sigma^+$, let $\theta = a_1 \cdots a_n$, $a_i \in \Sigma$, $i = 1, 2, ..., n$, $\phi_1(\theta) = \phi_1(a_1) \cdots \phi_1(a_n)$, $\phi_2(\theta) = \phi_2(a_1) \bullet \cdots \bullet \phi_2(a_n)$, then

$$\begin{aligned}
L(M_1)(\theta) &= \bigvee_{\gamma \in \Gamma^*} \nabla_1^*(([q_0, \varepsilon], \theta, Z_0), ([q_f, \varepsilon], \varepsilon, \gamma)) \\
&= \bigvee_{\gamma_i \in \Gamma^*, q_i \in Q_2} [\nabla_1(([q_0, \varepsilon], a_1 \cdots a_n, Z_0), ([q_0, \phi_1(a_1)], a_2 \cdots a_n, Z_0)) \\
&\bullet \nabla_1(([q_0, \phi_1(a_1)], \varepsilon a_2 \cdots a_n, Z_0), ([q_1, \varepsilon], a_2 \cdots a_n, \gamma_1))] \\
&\bullet [\nabla_1(([q_1, \varepsilon], a_2 \cdots a_n, \gamma_1), ([q_1, \phi_1(a_2)], a_3 \cdots a_n, \gamma_1)) \\
&\bullet \nabla_1(([q_1, \phi_1(a_2)], \varepsilon a_3 \cdots a_n, \gamma_1), ([q_2, \varepsilon], a_3 \cdots a_n, \gamma_2))] \bullet \cdots \\
&\bullet [\nabla_1(([q_{n-1}, \varepsilon], a_n, \gamma_{n-1}), ([q_{n-1}, \phi_1(a_n)], \varepsilon, \gamma_{n-1})) \\
&\bullet \nabla_1(([q_{n-1}, \phi_1(a_n)], \varepsilon, \gamma_{n-1}), ([q_f, \varepsilon], \varepsilon, \gamma))] \\
&= \bigvee_{\gamma, \gamma_i \in \Gamma^*, q_i \in Q_2} [\delta_1([q_0, \varepsilon], a_1, Z_0)([q_0, \phi_1(a_1)], Z_0) \\
&\bullet \delta_1([q_0, \phi_1(a_1)], \varepsilon, Z_0)([q_1, \varepsilon], \gamma_1)] \\
&\bullet [\delta_1([q_1, \varepsilon], a_2, \gamma_1)([q_1, \phi_1(a_2)], \gamma_1) \\
&\bullet \delta_1([q_1, \phi_1(a_2)], \varepsilon, \gamma_1)([q_2, \varepsilon], \gamma_2)] \bullet \cdots \\
&\bullet [\delta_1([q_{n-1}, \varepsilon], a_n, \gamma_{n-1})([q_{n-1}, \phi_1(a_n)], \gamma_{n-1}) \\
&\bullet \delta_1([q_{n-1}, \phi_1(a_n)], \varepsilon, \gamma_{n-1})([q_f, \varepsilon], \gamma)] \\
&= \bigvee_{\gamma, \gamma_i \in \Gamma^*, q_i \in Q_2} [\nabla_2^*((q_0, \phi_1(a_1), Z_0), (q_1, \varepsilon, \gamma_1)) \\
&\bullet \nabla_2^*((q_1, \phi_1(a_2), \gamma_1), (q_2, \varepsilon, \gamma_2)) \bullet \cdots \\
&\bullet \nabla_2^*((q_{n-1}, \phi_1(a_n), \gamma_{n-1}), (q_f, \varepsilon, \gamma))] \\
&\bullet [\phi_2(a_1) \bullet \cdots \bullet \phi_2(a_n)] \\
&= \bigvee_{\gamma, \gamma_i \in \Gamma^*, q_i \in Q_2} [\nabla_2^*((q_0, \phi_1(a_1)\phi_1(a_2) \cdots \phi_1(a_n), Z_0), (q_1, \varepsilon\phi_1(a_2) \cdots \phi_1(a_n), \gamma_1)) \\
&\bullet \nabla_2^*((q_1, \phi_1(a_2)\phi_1(a_3) \cdots \phi_1(a_n), \gamma_1), (q_2, \varepsilon\phi_1(a_3) \cdots \phi_1(a_n), \gamma_2)) \bullet \cdots \\
&\bullet \nabla_2^*((q_{n-1}, \phi_1(a_n), \gamma_{n-1}), (q_f, \varepsilon, \gamma))] \bullet \phi_2(\theta)
\end{aligned}$$

$$= \bigvee_{\gamma \in \Gamma^*} [\nabla_2^*((q_0, \phi_1(a_1) \cdots \phi_1(a_n), Z_0), (q_f, \varepsilon, \gamma))] \bullet \phi_2(\theta)$$
$$= f(\phi_1(\theta)) \bullet \phi_2(\theta)$$
$$= \phi^{-1}(f)(\theta).$$

Thus, $L(M_1) = \phi^{-1}(f)$.

Clearly, the second solution is much more efficient than the first one. Let $|Q_2|, |\Sigma|$ represent the number of their elements, respectively. For any $a_i \in \Sigma, \phi_1(a_i) \in \Delta^*$, $|\phi_1(a_i)|$ represents the number of its suffixes. Then we need $m = |Q_2| \times |\Sigma| \times \prod_{a_i \in \Sigma} |\phi_1(a_i)|$ steps in the proof I, while we just need $n = |Q_2| \times |\Sigma|$ steps in the proof II, that is, $n \ll m$, which means that the improved proof has greatly improved the efficiency of operation. From two different proofs above, it can be seen that the construction of new state sets is a particularly essential step and the key to improve the efficiency.

6 Conclusion

In this paper, we have studied fuzzy context-free grammars (FCFG), fuzzy context-free languages(FCFL) and fuzzy pushdown automata(FPDA) whose condomain forms a lattice-ordered monoid $\mathcal{L}$. Some important conclusions are obtained. Particularly, if $\mathcal{L}$ is commutative, we show an improved proof of the closure under homomorphic inverse of fuzzy context-free languages, which has greatly improved the efficiency of the operation by comparing with the classical method. Undoubtedly, much more work remains to be completed along this line. There may be more special properties of fuzzy context-free languages on lattice-ordered monoids.

References

1. Alasdair, U.: Balbes Raymond and Dwinger Philip. Distributive lattices. University of Missouri Press, Columbia 1974, xiii + 294 pp. J. Symb. Logic **42**, 294–588 (1997)
2. Asveld, P.R.J.: Algebraic aspects of families of fuzzy languages. Theoret. Comput. Sci. **293**(2), 417–445 (2003)
3. Asveld, P.R.J.: Fuzzy context-free languages part 1: generalized fuzzy context-free grammars. Theoret. Comput. Sci. **347**(1), 167–190 (2005)
4. Depalma, G.F., Yau, S.S.: Fractionally fuzzy grammars with application to pattern recognition. In: Fuzzy Sets & Their Applications to Cognitive & Decision Processes, pp. 329–351 (1975)
5. Dilworth, R.P., Birkhoff, G.: Lattice theory. Bull. Am. Math. Soc. **56**, 204–206 (1950)
6. Gerla, G.: Fuzzy grammars and recursively enumerable fuzzy languages. Inf. Sci. **60**(1C2), 137–143 (1992)
7. Guo, X.: Grammar theory based on lattice-ordered monoid. Fuzzy Sets Syst. **160**(8), 1152–1161 (2009)
8. Guo, X.: A comment on automata theory based on complete residuated lattice-valued logic: pushdown automata (2012)

9. Jiang, Z., Jiang, S.: Formal languages and automata. Tsinghua University Press (2003)
10. Jin, J., Li, Q.: Fuzzy grammar theory based on lattices. Soft Comput. **16**(8), 1415–1426 (2012)
11. Kim, H.H., Mizumoto, M., Toyoda, J., Tanaka, K.: L -fuzzy grammars. Inf. Sci. **8**(2), 123–140 (1975)
12. Lan, S.: A machine accepted fuzzy context-free languages and fuzzy pushdown automata. BUSEFAL **49**(6), 67–72 (1992)
13. Lee, E.T., Zadeh, L.A.: Note on fuzzy languages. Inf. Sci. **1**(4), 421–434 (1969)
14. Lee, E.T., Zadeh, L.A.: Fuzzy languages and their acceptance by automata. In: Fourth Princeton Conference Information Science and System, no. 2, pp. 399–410 (1970)
15. Li, P., Li, Y.M.: Algebraic properties of la-languages. Inf. Sci. **176**(21), 3232–3255 (2006)
16. Li, Y., Li, P.: Fuzzy computing theory. Science Press (2016)
17. Li, Y., Pedrycz, W.: Fuzzy finite automata and fuzzy regular expressions with membership values in lattice-ordered monoids. Fuzzy Sets Syst. **156**(25), 68–92 (2005)
18. Li, Z., Li, P., Li, Y.: The relationships among several types of fuzzy automata. Inf. Sci. **176**(15), 2208–2226 (2006)
19. Omlin, C.W., Giles, C.L.: Equivalence in knowledge representation: automata, recurrent neural networks, and dynamical fuzzy systems. Proc. IEEE **87**(9), 1623–1640 (1999)
20. Pan, H., Song, F., Cao, Y., Qian, J.: Fuzzy pushdown termination games. IEEE Trans. Fuzzy Syst. **27**(15), 760–774 (2019)
21. Qiu, D.: Automata theory based on complete residuated latticed-valued logic(i). Sci. China (Series E) **33**(10), 137–146 (2003)
22. Santos, E.: Fuzzy automata and languages. Inf. Sci. **10**(3), 193–197 (1976)
23. Santos, E.S.: Maximin automata. Inf. Control **13**(4), 363–377 (1968)
24. Schtzenberger, M.: On context-free languages and push-down automata. Inf. Control **6**(3), 246–264 (1963)
25. Senay, H.: Fuzzy command grammars for intelligent interface design. IEEE Trans. Syst. Man Cybern. **22**(5), 1124–1131 (1992)
26. Steimann, F., Adlassnig, K.P.: Clinical monitoring with fuzzy automata. Fuzzy Sets Syst. **61**(1), 37–42 (1994)
27. Wang, H., Qiu, D.: Computing with words via turing machines: a formal approach. IEEE Trans. Fuzzy Syst. **11**(6), 742–753 (2003)
28. Xing, H.: Fuzzy pushdown automata. Fuzzy Sets Syst. **158**(13), 1437–1449 (2007)
29. Xing, H., Qiu, D.: Automata theory based on complete residuated lattice-valued logic: a categorical approach. Fuzzy Sets Syst. **160**(16), 2416–2428 (2009)