# Autômatos Lineares *Fuzzy*

Valdigleis S. Costa[a,c] and Benjamín R. C. Bedregal[b,c]

[a]Programa de Pós-graduação em Sistemas e Computação - PPgSC
[b]Departamento de Informática e Matemática Aplicada - DIMAp
[c]Universidade Federal do Rio Grande do Norte - UFRN
valdigleis@ppgsc.ufrn.br,
bedregal@dimap.ufrn.br

**Resumo** As linguagens lineares *fuzzy* são conjuntos *fuzzy* gerado a partir das regras descritas por gramáticas lineares *fuzzy*, neste trabalho introduziremos duas novas formas normais para as gramática lineares *fuzzy*. Além disso, apresentaremos um novo método recursivo baseado em t-normas e t-conormas para calcular a pertinência com que uma gramática linear *fuzzy* é capaz de derivar uma determinada palavra. Em seguida, nós introduzimos os autômatos lineares *fuzzy*. Por fim, provaremos por meio da relação entre as gramáticas lineares *fuzzy* e autômatos lineares *fuzzy*, que os autômatos lineares *fuzzy* são dispositivos reconhecedores para a classe das linguagens lineares *fuzzy*.

**Palavras chaves:** Formas normais, Gramáticas lineares *fuzzy*, Autômatos lineares *fuzzy*

## 1 Introdução

A precisão das linguagens formais contrasta bastante com a imprecisão das linguagens naturais [1], então Lee e Zadeh em [2] introduziram as linguagens *fuzzy* como uma alternativa para aproximar as duas famílias de linguagens (formais e naturais). As linguagens *fuzzy* são organizadas usando a versão *fuzzy* da hierarquia de Chomsky [3] em quatro classes: recursivamente enumeráveis *fuzzy* ($L_{RE}$), sensíveis ao contexto *fuzzy* ($L_{SC}$), livres do contexto *fuzzy* ($L_{LC}$) e regulares *fuzzy* ($L_{Reg}$). Cada uma das classes de linguagens *fuzzy* possui um tipo específico de gramática *fuzzy* geradora, por exemplo, as linguagens pertencentes a $L_{Reg}$ são geradas pelas gramáticas *fuzzy* tipo 3 [1]. Além disso, cada classe apresenta no mínimo um modelo de máquina de estado *fuzzy* como seu dispositivo reconhecedor, na Tabela 1 listamos as classes de linguagens *fuzzy*, seu tipo de gramática geradora e um exemplo de dispositivo reconhecedor.

Nos últimos anos as gramáticas *fuzzy* e linguagens *fuzzy* vem sendo investigados por interesses teóricos e práticos, como podemos ver nos trabalhos [4][5][6][7][8][9][10]. Como dito em [11] outras classes de linguagens podem ser adicionadas a hierarquia de Chomsky, criado a hierarquia estendida de Chomsky. Uma dessas classes é a classe das linguagens lineares [12], tal classe possui como visto em [6] uma versão *fuzzy*, as linguagens pertencentes a classe das linguagens lineares *fuzzy* ($L_{lin}$) são aquela geradas por gramáticas lineares *fuzzy* [6].

| Classes | Gramáticas *Fuzzy* geradora | Máquinas reconhecedoras |
|---|---|---|
| $L_{RE}$ | Tipo 0 | Máquina de Turing *fuzzy* |
| $L_{SC}$ | Tipo 1 | Máquina de Turing *fuzzy* linearmente limitada |
| $L_{LC}$ | Tipo 2 | Autômato de pilha *fuzzy* |
| $L_{Reg}$ | Tipo 3 | Autômato *fuzzy* |

**Tabela 1.** Relação entre linguagens, gramáticas e máquinas de estado *Fuzzy*.

Diferente da teoria "clássica" das linguagens formais e dos autômatos, onde temos no mínimo quatro dispositivos reconhecedores [13][14][15][16] para a classe das linguagens lineares, não existe até o presente momento nenhuma máquina reconhecedora específica para $L_{lin}$. Assim neste trabalho apresentamos uma extensão *fuzzy* do modelo de $\lambda$-ALN proposto por Bedregal[15][17] para atuar como um reconhecedor para a classe $L_{lin}$. Além disto, apresentaremos duas novas formas normais para gramáticas lineares *fuzzy*.

Assim este trabalho estará organizado da seguinte forma: na Seção 1 esta introdução, na Seção 2 apresentaremos os pontos preliminares necessários ao trabalho, na Seção 3 apresentaremos as novas formas normais para as gramáticas lineares *fuzzy*, na Seção 4 definiremos os autômatos lineares fuzzy e investigaremos sua relação com as gramáticas lineares fuzzy. Por fim, na Seção 5 apresentaremos a conclusão e alguns apontamentos sobre o trabalho.

## 2 Preliminares

Nesta seção discutiremos algumas noções básicas que serão de suma importância para este trabalho.

### 2.1 Teoria *Fuzzy*

A Teoria *fuzzy* teve inicio com a ideia de conjunto *fuzzy*, que foi apresentada pela primeira vez por Zadeh em seu trabalho [18], formalmente definimos um conjunto *fuzzy* da seguinte forma.

**Definição 1** *[18]: Seja $\mathbb{U}$ um conjunto universo não vazio. Um conjunto fuzzy $\boldsymbol{A}$ no universo $\mathbb{U}$ é caracterizado pela função de pertinência $\mu_{\boldsymbol{A}} : \mathbb{U} \to [0,1]$, sendo $\mu_{\boldsymbol{A}}(x)$ o grau de pertinência do elemento $x$ no conjunto fuzzy $\boldsymbol{A}$, para todo $x \in \mathbb{U}$.*

Ao se trabalhar com as ideias apresentadas na teoria *fuzzy* é comum empregar os operadores t-normas e t-conormas [19].

**Definição 2** *Uma função $\otimes : [0,1]^2 \to [0,1]$ é uma t-norma (ou norma triangular) se satisfaz as seguintes propriedades:*

*P1. Comutatividade $\otimes(x,y) = \otimes(y,x)$;*
*P2. Associatividade $\otimes(x, \otimes(y,z)) = \otimes(\otimes(x,y),z)$;*
*P3. Monotonicidade, se $x \leq x'$ e $y \leq y'$ então $\otimes(x,y) \leq \otimes(x',y')$ e*

*P4. 1-Identidade (ou 1 como elemento neutro)* $\otimes(x,1)=x$.

**Definição 3** *Uma função* $\oplus : [0,1]^2 \to [0,1]$ *é uma t-conorma (ou conorma triangular) se satisfaz as propriedades P1, P2 e P3 (apresentados na definição de t-norma) e também P4 trocando o elemento neutro para 0.*

Por definição, as t-normas são operações binárias, no entanto, é interessante e conveniente estender esta operação para uma aridade $n$, essa extensão é feita com exposto em [19] da seguinte forma:

$$\bigotimes_{i=1}^{n} x_i = \otimes\left(\bigotimes_{i=1}^{n-1} x_i, x_n\right)$$

O fato das t-normas serem associativas, garante que essa extensão esta bem definida (para maiores detalhes ver [20]). E de forma dual podemos estender as t-conormas para operações de aridade $n$ apenas trocando $\otimes$ por $\oplus$ na equação acima, como visto em [19]. Salientamos que, no caso particular de $n=1$ temos que $\otimes(x)=x$ e $\oplus(x)=x$, além disso, estamos assumindo que as t-normas não possuem divisores de zero.

### 2.2 Linguagens e Gramáticas Lineares *Fuzzy*

Do trabalho inicial de Lee e Zadeh [2] nos temos que, uma Linguagem *fuzzy* $\mathbb{L}$, é um conjunto *fuzzy* sobre $\Sigma^*$. Assim $\mathbb{L}$ é o conjunto dos pares ordenados $\mathbb{L} = \{(x, \mu_{\mathbb{L}}(x)) : x \in \Sigma^*\}$. Como dito em [1], uma gramática *fuzzy* é um conjunto de regras capazes de gerar os elementos de um conjunto *fuzzy*, assim uma **linguagem linear *fuzzy*** é o conjunto *fuzzy* gerado por uma **gramática linear *fuzzy***.

**Definição 4** *[6]Uma gramática fuzzy é uma quadrupla* $\mathcal{G} = \langle V, \Sigma, \hat{S}, \hat{P}\rangle$ *onde:*

$V$ *é o conjunto de variáveis;*
$\Sigma$ *é o conjunto alfabeto*[1]*;*
$\hat{S} : V \to [0,1]$ *é um conjunto fuzzy de variáveis iniciais;*
$\hat{P} : V \times (V \cup \Sigma)^* \to [0,1]$ *é um conjunto fuzzy de produções.*

Assim como em [1][2][6] para simplificar, $\hat{P}(\alpha,\beta)=\sigma$ é escrito como $\alpha \xrightarrow{\sigma} \beta$ para algum $\sigma \in [0,1], \alpha \in V$ e $\beta \in (V \cup \Sigma)^*$. Uma gramática *fuzzy* $\mathcal{G}$ **será linear** se em todas as produções $\beta = x_1 A x_2$, para algum $A \in V \cup \{\lambda\}$ e $x_1, x_2 \in \Sigma^*$. Dizemos que uma palavra $w$ deriva diretamente uma palavra $w'$ em uma gramática linear *fuzzy*, denotado como $w \Rightarrow w'$, se $w = x\alpha y, w' = x\beta y$ e tem-se $\alpha \xrightarrow{\sigma} \beta \in \hat{P}$ com $x, y \in \Sigma^*$, a seguir apresentamos nossa interpretação de como se obter o grau de pertinência da derivação direta de uma palavra para outra.

[1] Destacamos que $V$ e $\Sigma$ são conjuntos disjuntos.

**Definição 5** *Dada uma gramática linear fuzzy $G$, o grau da derivação $w \Rightarrow w'$ e dado pela função $d : (V \cup \Sigma)^+ \times (V \cup \Sigma)^* \to [0,1]$ onde $d$ é definida como:*

$$d(w,w') = \begin{cases} \sigma, & se\ w = x\alpha y, w' = x\beta y, \alpha \xrightarrow{\sigma} \beta \in \hat{P}\ com\ x,y \in \Sigma^* \\ 0, & senão \end{cases} \quad (1)$$

Se $w_1, w_2, \cdots, w_n \in (V \cup \Sigma)^*$ e temos a seguinte sequência de derivações entre as palavras: $w_1 \Rightarrow w_2$, $w_2 \Rightarrow w_3$, $\cdots$, $w_{n-1} \Rightarrow w_n$, podemos então dizer que $w_1$ deriva $w_n$. Está derivação de $w_1$ para $w_n$ é chamada de cadeia de derivação. Existem diferentes métodos para se determinar a pertinência de uma derivação de $w_1$ para $w_n$ (ver [1][6][21]), a seguir apresentamos um novo método recursivo baseado no uso de t-normas e t-conormas.

**Definição 6** *Seja $w_n$ uma palavra derivável de $w_1$ por uma ou mais cadeias de derivação, então o valor de pertinência da derivação de $w_1$ para $w_n$ é dado pela função $\hat{d} : (V \cup \Sigma)^+ \times (V \cup \Sigma)^* \to [0,1]$, onde $\hat{d}$ é definida como:*

$$\hat{d}(w_1,w_n) = \begin{cases} d(w_1,w_n), & se\ w_1 \Rightarrow w_n \\ \bigoplus\limits_{w' \in (V \cup T)^+} \Big(\otimes(\hat{d}(w_1,w'), d(w',w_n))\Big), & senão \end{cases} \quad (2)$$

Dizemos que uma palavra $w$ é gerada por uma gramática linear *fuzzy* $\mathcal{G}$ se houver uma ou mais cadeias de derivação $\alpha$ para $w$, onde $\alpha$ é uma variável inicial *fuzzy*, formalmente definimos isto pela equação:

$$\mu_{\mathcal{G}}(w) = \bigoplus_{\alpha \in V, S(\alpha) > 0} \Big(\otimes(S(\alpha), \hat{d}(\alpha, w))\Big) \quad (3)$$

**Definição 7** *A linguagem fuzzy gerada por uma gramática linear fuzzy $\mathcal{G}$ é o conjunto, $L(\mathcal{G}) = \{(w, \mu_{\mathcal{G}}(w)) : w \in \Sigma^*\}$.*

## 3 Novas Formas Normais

Em [1][6] encontramos a noção de forma normal para gramáticas *fuzzy*, no entanto, esta forma normal estava direcionada apenas para a classe das gramáticas regulares *fuzzy*, já em [21] temos a ideia de forma normal para gramáticas livre do contexto *fuzzy*. Nós agora, propomos duas novas formas normais para a classe das gramáticas lineares *fuzzy*, para introduzi-las precisamos antes considerar a Definição 8 e Lema 1 a seguir.

**Definição 8** *Produções da forma $A \xrightarrow{\sigma} C$ tal que $A, C \in V$ são chamadas de produções unitárias.*

**Lema 1** *Se $\mathcal{G}$ é uma gramática linear fuzzy, então existe uma gramática linear fuzzy sem produções unitárias $\mathcal{G}'$ tal que $L(\mathcal{G}) = L(\mathcal{G}')$.*

*Demonstração.* Suponha que $\mathcal{G} = \langle V, \Sigma, \hat{S}, \hat{P}\rangle$ seja uma gramática linear *fuzzy* com produções unitárias, então construa $\mathcal{G}' = \langle V, \Sigma, \hat{S}, \hat{P}'\rangle$ através do seguinte algoritmo:

1. Inicialmente faça $\hat{P}' = \hat{P}$;
2. Para todo $A \in V$ faça:
   2.2 Enquanto existirem $A \xrightarrow{\sigma} B$ em $\hat{P}'$ onde temos que $B \in V$ com $A \neq B$ faça:
      2.2.1 Se $set_B = \{w : B \xrightarrow{\sigma'} w \text{ e } w \in (V \cup \Sigma)^*\} \neq \emptyset$. Então para cada $w \in set_B$ adicione em $\hat{P}'$ uma produção $A \xrightarrow{\sigma''} w$ tal que $\sigma'' = \otimes(\sigma, \sigma')$.
      2.2.2 Remova $A \xrightarrow{\sigma} B$ de $\hat{P}'$.
3. Para todo $A \in V$ se $A \xrightarrow{\sigma} A \in \hat{P}$ tal que $\sigma > 0$, então remova $A \xrightarrow{\sigma} A$ de $\hat{P}'$.

Para mostrar que $L(\mathcal{G}) = L(\mathcal{G}')$ devemos mostrar que o algoritmo de construção de $\mathcal{G}'$ não altera o valor de pertinência das derivações, assim suponha que $A, B \in V$ e $z \in (V \cup \Sigma)^*$ e que existem em $\hat{P}$ as produções $A \xrightarrow{\sigma} B$ e $B \xrightarrow{\sigma'} z$, assim temos em $\mathcal{G}$ uma cadeia de derivação com o seguinte grau de pertinência:

$$(I)\ \hat{d}(A, z) = \bigoplus \Big(\otimes(\hat{d}(A, B), d(B, z))\Big) = \bigoplus (\otimes(d(A, B), \sigma'))$$

Porém, por $\hat{P}$ ser um conjunto *fuzzy* temos que $B \xrightarrow{\sigma'} z$ é único e por definição como existe uma derivação direta $A \Rightarrow B$ com grau $\sigma$ isso implica que $\bigoplus (\otimes(d(A, B), \sigma')) = \bigoplus (\otimes(\sigma, \sigma')) = \otimes(\sigma, \sigma')$, por outro lado, como temos $A \xrightarrow{\sigma} B, B \xrightarrow{\sigma'} z \in \hat{P}$ então pelo algoritmo de construção de $\mathcal{G}'$ nós teremos que $A \xrightarrow{\sigma''} z \in \hat{P}'$, assim em $\mathcal{G}'$ temos uma derivação direta de $A \Rightarrow z$ com o grau de pertinência dado por:

$$(II)\ \hat{d}(A, z) = d(A, z) = \sigma''$$

Note que, pelo algoritmo de construção de $\mathcal{G}'$ temos que $\sigma'' = \otimes(\sigma, \sigma')$, logo $(I)$ é igual a $(II)$, isto é, o grau de pertinência da derivação é preservado. Assim para todo $(x, \sigma) \in L(\mathcal{G})$ tem-se que $\mu_{\mathcal{G}}(x) = \mu_{\mathcal{G}'}(x)$, logo $L(\mathcal{G}) = L(\mathcal{G}')$.

Uma variável $A \in V$ em uma gramática linear *fuzzy* será dita **linear à esquerda** se todas as sua produções são da forma $A \xrightarrow{\sigma} Ba$, e será dita **linear à direita** se todas as sua produções são da forma $A \xrightarrow{\sigma} aB$ para algum $B \in V \cup \{\lambda\}$ e $a \in \Sigma^*$.

As gramáticas lineares *fuzzy* vão estar na **forma normal linear** se todas as suas variáveis são lineares à esquerda ou lineares à direita, assim não vai existir uma variável que seja simultaneamente linear em ambas as "direções". Salientamos que diferente da forma normal em [6], a forma normal linear não necessita que a gramática seja estrita (à esquerda ou à direita).

**Teorema 1** *Para toda gramática linear fuzzy $G$. Existe uma gramática linear fuzzy $\mathcal{G}'$ na fornal normal linear tal que $L(\mathcal{G}) = L(\mathcal{G}')$.*

*Demonstração.* Sem perda de generalidade, assuma que $\mathcal{G} = \langle V, \Sigma, \hat{S}, \hat{P} \rangle$ não contém produções unitárias (Lema 1). Assim construiremos $\mathcal{G}' = \langle V', \Sigma, \hat{S}, \hat{P}' \rangle$ seguindo o algoritmo descrito a seguir.

1. Inicialmente faça $V' = V$ e $\hat{P}' = \hat{P}$.
2. Então para cada $A \in V, u \in \Sigma^+$ construa o conjunto $A_u = \{A \xrightarrow{\sigma} uBv :$ para algum $B \in V, v \in \Sigma^+\}$. Se $A_u \neq \emptyset$ então substitua cada produção $A \xrightarrow{\sigma} uBv \in A_u$ pelas produções $A \xrightarrow{1} uC$ e $C \xrightarrow{\sigma} Bv$ no conjunto $\hat{P}'$, onde $C$ será uma nova variável adicionada a $V'$. Ao fim do loop deste processo, todas as produções em $P'$ serão da forma:

$$A \xrightarrow{\sigma} uB, A \xrightarrow{\sigma} Bu \text{ ou } A \xrightarrow{\sigma} u$$

   para algum $A, B \in V'$ e $u \in \Sigma^*$.
3. Se existe $A \in V'$ que é linear à esquerda e linear à direita simultaneamente, então remova as produções da forma $A \xrightarrow{\sigma} Bv$ de $P'$ e adicione as produções $A \xrightarrow{1} C$ e $C \xrightarrow{\sigma} Bv$ tal que $C$ é uma nova variável adicionada a $V'$.

Para mostrar que $L(\mathcal{G}) = L(\mathcal{G}')$ como antes devemos mostrar que o algoritmo não alterar o grau de pertinência das derivações. Primeiro perceba que o passo 2 preserva a pertinência das derivações, visto que, se em $\mathcal{G}$ temos de forma direta que,

$$\hat{d}(A, uBv) = d(A, uBv) = \sigma$$

Pela a aplicação do passo 2 existem em $\hat{P}'$ as produções $A \xrightarrow{1} uC$ e $C \xrightarrow{\sigma} Bv$, que na gramática $\mathcal{G}'$ serão usadas na cadeia de derivação de $A$ para $uBv$ e tal cadeia apresenta o seguinte grau de pertinência:

$$\hat{d}(A, uBv) = \bigoplus \Big( \otimes (\hat{d}(A, uC), d(uC, uBv)) \Big) = \bigoplus (\otimes (d(A, uC), \sigma)) = \bigoplus (\otimes (1, \sigma)) = \sigma$$

Não alterando assim o valor de derivação. O mesmo ocorre também no passo 3 do algoritmo, pois, suponha que após a aplicação do passo 2 a variável $A$ tenha se tornado simultaneamente linear à esquerda e à direita, implicando que em $\mathcal{G}$ temos de forma direta que,

$$\hat{d}(A, Bv) = d(A, Bv) = \sigma$$

Já em $\mathcal{G}'$ pela aplicação do passo 3 teremos as produções $A \xrightarrow{1} C$ e $C \xrightarrow{\sigma} Bv$ e assim a cadeia de $A$ para $Bv$ preserva o grau de pertinência, como vemos abaixo:

$$\hat{d}(A, Bv) = \bigoplus \Big( \otimes (\hat{d}(A, C), d(C, Bv)) \Big) = \bigoplus (\otimes (1, \sigma)) = \sigma$$

Como o algoritmo não altera o valor de pertinência das derivação então para qualquer que seja $x \in \Sigma^*$ teremos que $\hat{d}(A, x)$ com $A \in V$ terá seu valor preservado e assim podemos concluir que $\mu_G(x) = \mu_{G'}(x)$, ou seja, $L(\mathcal{G}) = L(\mathcal{G}')$.

Por sua vez, para estar na **forma normal linear forte** uma gramática linear *fuzzy* necessita que todas as suas produções sejam da forma $A \xrightarrow{\sigma} aB$ ou $A \xrightarrow{\sigma} Ba$ para algum $A \in V$, $B \in V \cup \{\lambda\}$ e $a \in \Sigma \cup \{\lambda\}$.

**Teorema 2** *Para toda gramática linear fuzzy $\mathcal{G}$. Existe uma gramática linear fuzzy $\mathcal{G}'$ na forma normal linear forte tal que $L(\mathcal{G}) = L(\mathcal{G}')$.*

*Demonstração.* Obtenha uma gramática linear *fuzzy* $\mathcal{G} = \langle V, \Sigma, \hat{S}, \hat{P}\rangle$ na forma normal linear em seguida, construa $\mathcal{G}' = \langle V', \Sigma, \hat{S}, \hat{P}'\rangle$ aplicando o algoritmo descrito a seguir.

1. Inicialmente faça $V' = V$ e $\hat{P}' = \hat{P}$.
2. Para todo $A \in V$ e $a \in \Sigma$, se $A^a = \{A \xrightarrow{\sigma} ya : |y| \geq 2$ ou $y \in \Sigma$ e $\sigma \in [0,1]\} \neq \emptyset$, então introduza uma nova variável $C$ em $V'$, em seguida remova de $\hat{P}'$ todas as produções que estão em $A^a$ e introduza as produções $A \xrightarrow{1} Ca$ e $C \xrightarrow{\sigma} y$, para cada $A \xrightarrow{\sigma} ya$ removido.
3. Para todo $A \in V$ e $a \in \Sigma$, se $A_a = \{A \xrightarrow{\sigma} ay : |y| \geq 2$ ou $y \in \Sigma$ e $\sigma \in [0,1]\} \neq \emptyset$, então introduza uma nova variável $C$ em $V'$, em seguida remova de $\hat{P}'$ todas as produções que estão em $A_a$ e introduza as produções $A \xrightarrow{1} aC$ e $C \xrightarrow{\sigma} y$, para cada $A \xrightarrow{\sigma} ay$ removido.

E para mostrar que $L(\mathcal{G}) = L(\mathcal{G}')$ basta mostrar que o algoritmo não alterar a pertinência das derivações, analisemos o caso do passo 2 (o mesmo é similar para o passo 3). Suponha que em $\mathcal{G}$ temos $A \in V$ e a produção $A \xrightarrow{\sigma} ya$ assim teremos de forma direta uma derivação de $A \Rightarrow ya$, assuma que $y \in \Sigma$ (o mesmo pode ser feito por indução no caso de $|y| \geq 2$), assim o grau de pertinência da derivação de $A$ para $ya$ é dado por:

$$\hat{d}(A, ya) = d(A, ya) = \sigma$$

E pelo algoritmo de construção de $\mathcal{G}'$ teremos em $\mathcal{G}'$, uma cadeia de derivação de $A$ para $ya$, que como podemos ver preserva o grau de pertinência.

$$\hat{d}(A, ay) = \bigoplus \Big(\otimes(\hat{d}(A, Ca), d(Ca, ya))\Big) = \bigoplus (\otimes(1, \sigma)) = \bigoplus (\sigma) = \sigma$$

Não alterado o valor da pertinência da derivação, e desta forma teremos para todo $(x, \sigma) \in L(\mathcal{G})$ que $\mu_{\mathcal{G}}(x) = \mu_{\mathcal{G}'}(x)$, portanto, $L(\mathcal{G}) = L(\mathcal{G}')$.

## 4 Autômatos Lineares *Fuzzy*

Usando o modelo de $\lambda$-ALN introduzido por Bedregal em [15] como base, nós definimos os autômatos lineares *fuzzy* como se segue.

**Definição 9** *Um autômato linear fuzzy é uma tupla $\mathcal{M} = \langle Q_1, Q_2, \Sigma, \mu, \hat{I}, F\rangle$ onde:*

*$Q_1$ e $Q_2$ são conjuntos disjuntos de estados;*
*$\Sigma$ é um conjunto de símbolos chamado alfabeto;*
*$\mu : (Q_1 \cup Q_2) \times (\Sigma \cup \{\lambda\}) \times (Q_1 \cup Q_2) \to [0,1]$ é um conjunto fuzzy;*
*$\hat{I} : (Q_1 \cup Q_2) \to [0,1]$ é um conjunto fuzzy de estados iniciais e*
*$F \subseteq (Q_1 \cup Q_2)$ é um conjunto de estados finais.*

Para facilitar nossa escrita mais adiante escreveremos $\mu(q, a, q')$ como sendo $(q, q')_a$. Os autômatos lineares *fuzzy* possuem as mesmas características em relação a "arquitetura" que os $\lambda$-ALN em [15][17], assim, ao trabalhar com autômatos lineares *fuzzy* podemos adotar a ideia de **descrição instantânea** (DI). Uma DI para um autômato linear *fuzzy* será representada da mesma forma que as DI's para um $\lambda$-ALN, isto é, será um par $(q, w) \in (Q_1 \cup Q_2) \times \Sigma^*$. Um passo de computação em um autômato linear *fuzzy* corresponde a mudança de uma DI para outra. Antes de definir formalmente o que seria um passo de computação, considere as seguintes funções.

**Definição 10** *Dado $w \in \Sigma^*$, tal que $w = x_1 x_2 \dots x_{n-1} x_n$ com $x_i \in \Sigma \cup \{\lambda\}$, definimos as funções $R_l, R_r, M_l, M_r : \Sigma^* \to \Sigma^*$ da seguinte forma.*

$$R_l(w) = \begin{cases} x_1, \text{ se } |w| > 0 \\ \lambda, \quad \text{senão} \end{cases} \qquad M_l(w) = \begin{cases} x_2 \dots x_{n-1} x_n, \text{ se } |w| > 1 \\ \lambda, \qquad\qquad \text{senão} \end{cases}$$

$$R_r(w) = \begin{cases} x_n, \text{ se } |w| > 0 \\ \lambda, \quad \text{senão} \end{cases} \qquad M_r(w) = \begin{cases} x_1 x_2 \dots x_{n-1}, \text{ se } |w| > 1 \\ \lambda, \qquad\qquad \text{senão} \end{cases}$$

Como já dito, um passo de computação em um autômato linear *fuzzy* $\mathcal{M}$ corresponde a mudança de uma DI $(q, w)$ para outra $(q', w')$ denotamos isto como $(q, w) \succ_{\mathcal{M}} (q', w')$. Um passo de computação de forma precisa consiste na aplicação de alguma instrução *fuzzy* sobre a DI de origem $(q, w)$ a fim de transforma-lá na DI de destino $(q', w')$, isto é, aplicar algum $(q, q')_a$ onde $q, q' \in (Q_1 \cup Q_2), a \in (\Sigma \cup \{\lambda\})$. Como cada instrução *fuzzy* tem associado a si um grau de pertinência de ser executada, podemos concluir que um passo de computação terá o mesmo grau de pertinência da instrução usada, a seguir definimos isto formalmente.

**Definição 11** *Dado um passo de computação $(q, w) \succ_{\mathcal{M}} (q', w')$ em um autômato linear fuzzy $\mathcal{M}$ o grau de pertinência de tal passo de computação e dado pela função $D : (Q_1 \cup Q_2) \times \Sigma^* \times (Q_1 \cup Q_2) \times \Sigma^* \to [0, 1]$ onde $D$ é definido como:*

$$D(q, w, q', w') = \begin{cases} (q, q')_\lambda, & \text{se } q \in Q_1 \cup Q_2, w' = w \\ (q, q')_{R_l(w)}, & \text{se } q \in Q_1, w' = M_l(w) \\ (q, q')_{R_r(w)}, & \text{se } q \in Q_2, w' = M_r(w) \end{cases} \tag{4}$$

Dizemos que $(q'', w'')$ é alcançável a partir de uma DI $(q, w)$, denotado por $(q, w) \succ^*_{\mathcal{M}} (q'', w'')$, se existir uma outra DI $(q', w')$ tal que $(q, w) \succ_{\mathcal{M}} (q', w')$ e $(q', w') \succ^*_{\mathcal{M}} (q'', w'')$, chamamos isso de cadeia de computação.

**Definição 12** *Seja $(q, w) \succ^*_{\mathcal{M}} (q'', w'')$ uma cadeia de computação, o grau de pertinência de tal cadeia é dado pela função $\hat{D} : (Q_1 \cup Q_2) \times \Sigma^* \times (Q_1 \cup Q_2) \times \Sigma^* \to [0, 1]$, onde $\hat{D}$ é definido como:*

$$\hat{D}(q, w, q'', w'') = \begin{cases} D(q, w, q'', w''), & \text{se } (q, w) \succ_{\mathcal{M}} (q'', w'') \\ \bigoplus\limits_{(q', w') \in Q_1 \cup Q_2 \times \Sigma^*} \left( \otimes (D(q, w, q', w'), \hat{D}(q', w', q'', w'')) \right), & \text{senão} \end{cases} \tag{5}$$

Sabe-se que um autômato reconhece uma palavra de entrada se partido de um estado inicial ele consome todos os símbolos da palavra de entrada e chega em um estado final. Podemos então definir este mesmo conceito para os autômatos lineares *fuzzy* da seguinte forma.

**Definição 13** *Seja $\mathcal{M} = \langle Q_1, Q_2, \Sigma, \mu, \hat{I}, F\rangle$ um autômato linear fuzzy e seja $w \in \Sigma^*$. Então o grau de pertinência de $\mathcal{M}$ reconhecer $w$ é dado pela seguinte equação:*

$$deg_{\mathcal{M}}(w) = \bigoplus_{\hat{I}(q)>0, q' \in F} \Big(\otimes(\hat{I}(q), \hat{D}(q, w, q', \lambda))\Big) \tag{6}$$

Assim a linguagem de qualquer autômato linear *fuzzy* $\mathcal{M}$, é o conjunto de todas as palavras reconhecidas pelos autômatos lineares *fuzzy*, formalmente temos:

$$L(\mathcal{M}) = \{(w, deg_{\mathcal{M}}(w)) : w \in \Sigma^*\} \tag{7}$$

Assim como na teoria "clássica" dos autômatos e das linguagens podemos ver a equivalência entre as linguagens reconhecidas por autômatos lineares *fuzzy* e as linguagens geradas por gramáticas lineares *fuzzy*, os dois próximos teoremas se preocupam exatamente em mostrar esta relação.

**Teorema 3** *Seja $\mathcal{M} = \langle Q_1, Q_2, \Sigma, \mu, \hat{I}, F\rangle$ um autômato linear fuzzy. Então existe uma gramática linear fuzzy $\mathcal{G} = \langle V, \Sigma, \hat{S}, \hat{P}\rangle$ tal que $L(\mathcal{G}) = L(\mathcal{M})$.*

*Demonstração.* Construímos a gramática linear *fuzzy* $\mathcal{G} = \langle Q_1 \cup Q_2, \Sigma, \hat{I}, \hat{P}\rangle$ com $\hat{P}$ sendo:

$$\begin{aligned}\hat{P} = &\{q' \xrightarrow{\sigma} aq'' : \mu(q', q'')_a = \sigma, q' \in Q_1, q'' \in (Q_1 \cup Q_2), a \in \Sigma \cup \{\lambda\}\} \cup \\ &\{q' \xrightarrow{\sigma'} q''a : \mu(q', q'')_a = \sigma', q' \in Q_2, q'' \in (Q_1 \cup Q_2), a \in \Sigma \cup \{\lambda\}\} \cup \\ &\{q' \xrightarrow{1} \lambda : q' \in F\}\end{aligned}$$

Agora para todo $(x, \sigma) \in L(\mathcal{M})$ com $x = a_1 a_2 \cdots a_{n-1} a_n$ tal que $a_i \in (\Sigma \cup \{\lambda\})$ tem-se que existe $q_0, q_n \in (Q_1 \cup Q_2)$ tal que $(q_0, w) \succ^*_{\mathcal{M}} (q_n, \lambda)$ com $\hat{I}(q_0) > 0, q_n \in F$ logo temos $deg_{\mathcal{M}}(x) = \bigoplus\{\otimes(\hat{I}(q_0), \hat{D}(q_0, x', q_n, \lambda))\} = \sigma$. Porém de $(q_0, w) \succ^*_{\mathcal{M}} (q_n, \lambda)$ implica que existem as instruções *fuzzy*: $(q_0, q_1)_{a_1} = \sigma_1, (q_1, q_2)_{a_2} = \sigma_2, \cdots, (q_{n-2}, q_{n-1})_{a_{n-1}} = \sigma_{n-1}, (q_{n-1}, q_n)_{a_n} = \sigma_n$. Portanto, para cada $(q_i, q_{i+1})_{a_{i+1}}$ com $0 \leq i \leq n-1$, existe uma produção *fuzzy* correspondente, que será da forma $q_i \xrightarrow{\sigma_{i+1}} a_{i+1} q_{i+1}$ sempre que $q_i \in Q_1$, ou então, da forma $q_i \xrightarrow{\sigma_{i+1}} q_{i+1} a_{i+1}$ sempre que $q_i \in Q_2$ e também vai existir uma produção $q_n \xrightarrow{1} \lambda$ para todo $q_n \in F$. Note que, $x$ pode ser visto como $x = y_1 y_2$ tal que $y_1, y_2 \in \Sigma^*$, agora perceba para todo $x$ pela construção de $\mathcal{G}$ temos que existi uma variável $q_0 \in V$ e uma palavra $x' = y_1 q_n y_2$, tal que existe uma derivação $\hat{d}(q_0, x')$ usando exatamente as produções geradas pelo algoritmo de construção de $\mathcal{G}$ a partir das instruções $(q_0, q_1)_{a_1} = \sigma_1, (q_1, q_2)_{a_2} = \sigma_2, \cdots, (q_{n-2}, q_{n-1})_{a_{n-1}} = \sigma_{n-1}, (q_{n-1}, q_n)_{a_n} = \sigma_n$ e como existe $q_n \xrightarrow{1} \lambda$, então $y_1 q_n y_2 \Rightarrow y_1 y_2$ com o grau de pertinência dado por, $d(y_1 q_n y_2, y_1 y_2) = 1$ e assim:

$$\begin{aligned}\hat{d}(q_0,x) &= \bigoplus\Big(\otimes(\hat{d}(q_0,x'),d(x',x))\Big)\\ &= \bigoplus\Big(\otimes(\hat{d}(q_0,y_1q_ny_2),d(y_1q_ny_2,y_1y_2))\Big)\\ &= \bigoplus\Big(\otimes(\hat{d}(q_0,y_1q_ny_2),1)\Big)\\ &= \bigoplus\Big(\hat{d}(q_0,x')\Big)\end{aligned}$$

Dessa forma, $\mu_{\mathcal{G}}(x) = \bigoplus\Big(\otimes(\hat{I}(q_0),\hat{d}(q_0,x'))\Big)$ e pela construção de $\mathcal{G}$ temos que $\hat{d}(q_0,x)$ possui uma produção *fuzzy* correspondente a cada instrução *fuzzy* $(q_i,q_{i+1})_{a_{i+1}}$ usando em $\hat{D}(q_0,x,q_n,\lambda)$, implicando que $\hat{d}(q_0,x') = \hat{D}(q_0,x,q_n,\lambda)$, assim temos que $\mu_{\mathcal{G}}(x) = \bigoplus\Big(\otimes(\hat{I}(q_0),\hat{D}(q_0,x',q_n,\lambda))\Big)$, no entanto, por definição $deg_{\mathcal{M}}(x) = \bigoplus\Big(\otimes(\hat{I}(q_0),\hat{D}(q_0,x',q_n,\lambda))\Big)$, logo $\mu_{\mathcal{G}}(x) = deg_{\mathcal{M}}(x)$ e assim $L(\mathcal{G}) = L(\mathcal{M})$.

E como para o caso clássico [15] também podemos estabelecer a "volta" do Teorema 3, e isto é feito a seguir.

**Teorema 4** *Se $\mathcal{G} = \langle V,\Sigma,\hat{S},\hat{P}\rangle$ é um gramática linear fuzzy. Então existe um autômato linear fuzzy $\mathcal{M} = \langle Q_1,Q_2,\Sigma,\mu,\hat{I},F\rangle$ tal que $L(\mathcal{M}) = L(\mathcal{G})$.*

*Demonstração.* Sem perda de generalidade assuma que a gramática $\mathcal{G}$ está na forma normal linear forte. Então considere $\mathcal{M} = \langle Q_1,Q_2,\Sigma,\mu,\hat{S},F\rangle$ com:

(a) $Q_1 = \{A : A \text{ é um variável linear à direita}\} \cup \{C\}$, onde $C \notin V$ e $Q_2 = \{A : A \text{ é um variável linear à esquerda}\}$;
(b) $\mu$ é definido para todo $A,B \in V$ e $a \in \Sigma \in \{\lambda\}$ da seguinte forma:

$$(A,B)_a = \sigma, \text{se } A \xrightarrow{\sigma} aB \text{ ou } A \xrightarrow{\sigma} Ba \in \hat{P}$$
$$(A,C)_a = \sigma, \text{se } A \xrightarrow{\sigma} a \in \hat{P}$$

(c) $F = \{C\}$.

Temos que para todo $(x,\sigma) \in L(\mathcal{G})$ com $x = a_1a_2\cdots a_{n-1}a_n$ tal que $a_i \in (\Sigma \cup \{\lambda\})$ implica que, $\mu_G(x) = \sigma$. Então vai existir $A_1 \in V$ e no mínimo uma derivação de $A_1$ para $x$, isto é, $\hat{d}(A_1,x) = \sigma$ com $\hat{S}(A_1) > 0$, tal que a $i$-ésima produção usada na cadeia de derivação é da forma $A_i \xrightarrow{\sigma_i} a_iA_{i+1}$(ou $A_i \xrightarrow{\sigma_i} a_iA_{i+1}$) com $A_i \in V, A_{i+1} \in V \cup \{\lambda\}, a_i \in \Sigma \cup \{\lambda\}$ e $\sigma_i \in [0,1]$, e por sua vez, a última produção usada na cadeia será da forma $A_n \xrightarrow{\sigma_n} a_n$ onde $A_n \in V, a_n \in \Sigma \cup \{\lambda\}$ e $\sigma_n \in [0,1]$. Pelo passo $(b)$ do algoritmo de construção de $\mathcal{M}$ para cada produção *fuzzy* teremos uma instruções *fuzzy* correspondente, tais instruções serão da forma: $(A_1,A_2)_{a_1} = \sigma_1, (A_2,A_3)_{a_2} = \sigma_2, \cdots, (A_{n-1},A_n)_{a_{n-1}} = \sigma_{n-1}, (A_n,C)_{a_n} = \sigma_n$ e pelo passo $(c)$ temos $C \in F$, assim é fácil ver que $(A,x) \succ^*_{\mathcal{M}} (C,\lambda)$ logo temos,

$$deg_{\mathcal{M}}(x) = \bigoplus\Big(\otimes(\hat{S}(A),\hat{D}(A,x,C,\lambda))\Big)$$

Porém, como cada instrução em $\mu$ corresponde a uma produção usada em $\hat{d}(A,x)$ e isto implica que, $\hat{D}(A,x,C,\lambda) = \hat{d}(A,x)$, substituindo na equação anterior temos,

$$deg_{\mathcal{M}}(x) = \bigoplus \Big( \otimes (\hat{S}(A), \hat{d}(A,x)) \Big)$$

No entanto, por definição $\mu_{\mathcal{G}}(x) = \bigoplus \Big( \otimes (\hat{S}(A), \hat{d}(A,x)) \Big)$ e assim temos que, $\deg_{\mathcal{M}}(x) = \mu_{\mathcal{G}}(x)$. Logo podemos concluir que $L(\mathcal{M}) = L(\mathcal{G})$.

Dado esta simetria na relação entre as linguagens reconhecidas pelos autômatos lineares *fuzzy* e as linguagens geradas pelas gramáticas lineares *fuzzy* podemos imediatamente pelos Teoremas 3 e 4 inferir o corolário a seguir, evidenciando que os autômatos lineares *fuzzy* são de fato dispositivos reconhecedores da classe $L_{lin}$.

**Corolário 1** *Uma linguagem $\mathbb{L}$ é linear fuzzy se, e somente se, existe um autômato linear fuzzy $\mathcal{M}$ tal que $L(\mathcal{M}) = \mathbb{L}$.*

## 5 Conclusão e Apontamentos

Neste artigo, introduzimos a forma normal linear e a forma normal linear forte para a classe das gramáticas lineares *fuzzy*. Além disto, propomos um novo tipo de autômato *fuzzy* que chamamos de autômato linear *fuzzy*, em seguida, provamos através da relação com as gramáticas lineares *fuzzy* que os autômatos lineares *fuzzy* são reconhecedores das linguagens lineares *fuzzy*, isto é, os autômatos lineares *fuzzy* são maquinas reconhecedoras para a classe $L_{lin}$. Infelizmente dado a quantidade mínima de paginas desse artigo não foi possível inserir exemplos do uso dos Teoremas aqui provados, em trabalhos futuros iremos inserir tais exemplos. Quanto a aplicações práticas e teóricas dos autômatos lineares *fuzzy* e das gramáticas lineares *fuzzy*, o trabalho [22] (realizado paralelamente a este), mostra tais aplicações.

## Agradecimentos

Agradecemos aos revisores anônimos pelas suas sugestões que ajudaram a melhorar a qualidade deste artigo e também a CAPES e a FAPERN pelo suporte financeiro oferecido ao primeiro autor.

## Referências

1. J. N. Mordeson and Davender S. Malik. *Fuzzy Automata and Languages: Theory and Applications*. Chapman & Hall/CRC, Washington, D. C., USA, 2002.
2. E. T. Lee and L. A. Zadeh. Note on Fuzzy Languages. *Information Sciences*, 1:421–434, 1969.
3. N. Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*, 2:113–124, 1956.

4. P. R. J. Asveld. Fuzzy context-free languages—Part 1: Generalized fuzzy context-free grammars. *Theoretical Computer Science*, 347:167–190, 2005.
5. P. R. J. Asveld. Algebraic aspects of families of fuzzy languages. *Theoretical Computer Science*, 293:417–445, 2003.
6. S. R. Chaudhari and D. D. Komejwar. On Fuzzy Regular Grammars. *Advances in Fuzzy Mathematics*, 6(1):89–104, 2011.
7. O. Unold. Learning Fuzzy Context-Free Grammar. In *Grammatical Inference: Theoretical Results and Applications*, pages 309–312, Berlin Heidelberg, 2010. Springer.
8. J. N. Mordeson and Davender S. Malik. On fuzzy recognizers. *Kybernetes*, 28(1):47–60, 1999.
9. M. Inui, W. Shoaff, L. Fausett, and M. Schneider. The recognition of imperfect strings generated by fuzzy context sensitive grammars. *Fuzzy Sets and Systems*, 62:21–29, 1994.
10. M. Schneider, H. Lim, and W. Shoaff. The utilization of fuzzy sets in the recognition of imperfect strings. *Fuzzy Sets and Systems*, 49(3):331–337, 1992.
11. B. R. C. Bedregal, B. M. Acióly, and A. Lyra. *Introdução à Teoria da Computação: Linguagens Formais, Autômatos e Computabilidade*. EdUnP/FAPERN, Natal, RN, 1ª edition, November 2010.
12. V. Amar and G. Putzolu. On a Family of Linear Grammars. *Information and Control*, 7:283–291, 1964.
13. M. A. Harrison. *Introduction to Formal Language Theory*. Addison-Wesley, London, UK, 1ª edition, 1978.
14. J. E. Hopcroft, R. Motwani, and J. D. Ullman. *Introduction To Automata Theory, Languages, and Computation*. Addison-Wesley, New York, USA, 2ª edition, 2001.
15. B. R. C. Bedregal. $\lambda$-ALN: Autômatos Lineares Não-Determinísticos com $\lambda$-Transições. *TEMA - Tendência em Matemática Aplicada e Computacional*, 12(3):171–182, 2011.
16. A.L. Rosenberg. A machine realization of the linear context-free languages. *Information and Control*, 10:175–188, 1967.
17. B. R. C. Bedregal. Nondeterministic Linear Automata and a Class of Deterministic Linear Languages. *Preliminary Proceedings LSFA*, pages 183–196, 2015.
18. L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.
19. E. P. Klement, Radko Mesiar, and Endre Pap. *Triangular Norms*, volume 8 of *Trends in Logic*. Springer-Science+Business Media, B.V., Poland, 2000.
20. I. A. Silva, B. R. C. Bedregal, and H. Bustince. Weighted Average Operators Generated by n-dimensional Overlaps and an Application in Decision Making. In *EUSFLAT*, 2015.
21. H. Xing, QIU Daowen, and Fuchun Liu. Automata theory based on complete residuated lattice-valued logic: Pushdown automata. *Fuzzy Sets and Systems*, 160(8):1125–1140, 2009.
22. V. da S. Costa. Linguagens lineares fuzzy. Master's thesis, Programa de Pós-graduação em Sistemas e Computação, UFRN, Natal, RN, Agosto 2016.