ISSN 1984-820X
Anais do CNMAC v.3
PUBLICAÇÃO DA SBMAC
CNMAC2010
XXXIII CONGRESSO NACIONAL DE MATEMÁTICA APLICADA E COMPUTACIONAL

# $\lambda$-ALN: Autômatos Lineares Não Determinísticos com $\lambda$-Transições

**Benjamín Callejas Bedregal**
Grupo de Lógica, Linguagem, Informação, Teoria e Aplicações – LoLITA,
Departamento de Informática e Matemática Aplicada – DIMAp,
Universidade Federal do Rio Grande do Norte – UFRN
Campus Universitário s/n, 59072-970, Natal, RN
E-mail: bedregal@dimap.ufrn.br

**Resumo:** *Neste trabalho introduzimos a classe dos autômatos lineares não-determinísticos com $\lambda$-transições e baseados numa forma normal, também aqui introduzida, para gramáticas lineares, provamos que a classe de linguagens aceita por este tipo de autômatos é exatamente a classe das linguagens lineares.*
**Palavras-chave:** *Linguagens lineares, gramáticas lineares, forma normal, autômatos lineares não determinísticos, $\lambda$-transições.*

## 1 Introdução

A classe das linguagens lineares se localiza dentro da hierarquia de Chomsky entre as linguagens regulares e as livres de contexto. O principal modelo computacional para esta classe de linguagens são as gramáticas lineares. Esta gramática permite gerar cadeias fazendo casamentos entre prefixos e sufixos de subcadeias da cadeia a ser gerada. Esta capacidade de casamentos faz com que esta classe de linguagens contenha a maioria das linguagens livres de contexto usuais. Assim, por exemplo a linguagem dos palíndromos e $a^nb^n$, que em geral são usadas como exemplos de linguagens livres de contexto que não são regulares, são na verdade linguagens lineares. Um exemplo de uma linguagem que é livre de contexto e que não é linear é: $\{a^nb^na^mb^m : n, m \geq 1\}$.

Em termos de autômatos, a classe das linguagens lineares têm ao menos três diferentes modelos: Um tipo especial de autômato finito de duas fitas [9, 4], tradutores finitos [9, 8] e autômatos a pilha com uma volta [4, 6, 2, 5]. Apesar dos méritos de cada um desses modelos, eles não são nem naturais nem intrínsecos, pois consideram elementos externos ao modelo. Por exemplo, no tipo especial de autômato finito de duas fitas a entrada é dividia nas duas fitas, de tal modo que a segunda fita contém o pedaço mais à direita da cadeia de entrada porém invertida. Note que os autômatos usuais (autômatos finitos determinísticos, autômatos a pilha, máquinas de Turing – mesmo a de duas fitas –, etc.) assumem que a entrada inteira está na fita de entrada ao momento de iniciar a computação. Assim, este modelo não possui esta característica natural. Além disso, a escolha de onde dividir assim como a reversão da parte direita é feita externa ao modelo. Portanto, este modelo não é nem natural nem intrínseco. O tradutor finito também divide a entrada em duas partes, mas as coloca numa única fita separadas por um símbolo especial, sendo que a segunda parte é invertida. Assim, analogamente ao modelo anterior, tradutores finitos não são nem intrínsecos nem naturais. Finalmente, uma vez que a volta no autômato a pilha é um movimento o qual decresce a pilha e que é precedido por outro que aumenta a pilha, o autômato a pilha com uma volta é claramente não natural. Como a restrição que o autômato pode fazer no máximo uma volta não é parte da estrutura do autômato (só analisamos a posteriori se o autômato satisfaz ou não esta restrição) este modelo tampouco é intrínseco.

Neste artigo propomos um modelo de autômatos alternativos para a classe das linguagens lineares que chamaremos *autômatos lineares não-determinísticos com $\lambda$-transições*, que em nosso ponto de vista é mais natural e simples que os mostrados anteriormente. Além disso nosso modelo é intrínseco e próprio, no sentido que não é uma restrição de algum tipo de autômato.

## Gramáticas Lineares

Como usual uma gramática formal $G$ é uma quádrupla $\langle V, T, S, P\rangle$ onde $V$ é um conjunto finito de símbolos variáveis, $T$ é um conjunto finito de símbolos terminais e portanto $V \cap T = \emptyset$, $S \in V$ é a variável de inicio e $P \subseteq (V \cup T)^+ \times (V \cup T)^*$ é o conjunto de produções. Pares ordenados $(x, y) \in P$ são denotados por $x \to y$.

**Definição 1.1** *A gramática $G = \langle V, T, S, P\rangle$ é **linear**, se cada produção em $P$ tem como lado esquerdo uma variável e como lado direito no máximo uma variável, sem qualquer restrição da posição dessa eventual variável. Uma variável $A \in V$ será chamada **linear à esquerda** se cada produção $A \to y \in P$ ou $y \in T^*$ ou $y = Bz$ para algum $z \in T^*$ e $B \in V$. Analogamente, uma variável $A \in V$ será chamada **linear à direita** se cada produção $A \to y \in P$ ou $y \in T^*$ ou $y = zB$ para algum $z \in T^*$ e $B \in V$. Uma gramática linear $G$ está na in **forma normal linear** (FNL), se cada variável $A \in V$ ou é linear à direita ou linear à esquerda.*

Note que uma variável $A$ numa gramática linear será linear à direita e linear à esquerda apenas quando todas as produções tendo $A$ no lado esquerdo tem seu lado direito só composto por símbolos terminais. Note também que nossa forma normal linear difere da forma normal linear usual (por exemplo em [10, 4, 7]) pois elas toleram duas produções com o mesmo lado esquerdo, mas com lados direitos tendo, numa delas, uma variável na posição mais à esquerda enquanto na outra a variável está na posição mais à direita, respectivamente.

**Exemplo 1.2** *A gramática $G = \langle\{S\}, \{a, b\}, S, P\rangle$ onde $P$ é definido por*

$$S \to aSbb \mid aSbbb \mid abb \mid abbb$$

*é linear mas não está na FNL.*

**Exemplo 1.3** *A gramática $G = \langle\{S, A, B, C\}, \{a, b\}, S, P\rangle$ onde $P$ é definido por*
$S \to aS \mid aA \mid B$
$A \to bAaa \mid \lambda$
$B \to Bb \mid Cb$
$C \to bbCa \mid \lambda$
*também é linear mas não está na FNL.*

Como usual, para cada $u, v, w \in (V \cup T)^*$ e $A \in V$, $uAw \Rightarrow uvw$ se existe uma produção $A \to v \in P$. Seja $\Rightarrow^*$ o fecho reflexivo e transitivo de $\Rightarrow$. A **linguagem gerada** por uma gramática $G$ é

$$L(G) = \{w \in T^* : S \Rightarrow^* w\}$$

No exemplo 1.2,

$$L(G) = \{a^m b^n : 1 \leq 2m \leq n \leq 3m\}$$

e no exemplo 1.3,

$$L(G) = \{a^m b^n a^{2n} : m \geq 1 \text{ e } n \geq 0\} \cup \{a^k b^{2m} a^m b^n : k, m \geq 0 \text{ e } n \geq 1\}. \tag{1}$$

Linguagens geradas por gramáticas lineares são chamadas **linguagens lineares**.

**Lema 1.4** *Seja $G$ uma gramática linear. Existe uma gramática linear $G'$ na FNL tal que $L(G) = L(G')$.*

**Exemplo 1.5** *A FNL de $G$ do exemplo 1.2 é a gramática $G' = \langle\{S, A\}, \{a, b\}, S, P'\rangle$ on $P'$ é definido por*
$S \to aA$
$A \to Sbb \mid Sbbb \mid bb \mid bbb$

*A FNL da gramática do exemplo 1.3 é a gramática* $G'' = \langle\{S, A, B, C, X, Y\}, \{a, b\}, S, P''\rangle$ *onde* $P''$ *é definida por*

$S \to aS \mid aA \mid B$
$A \to bX \mid \lambda$
$X \to Aaa$
$B \to Bb \mid Cb$
$C \to bbY \mid \lambda$
$Y \to Ca$

Uma gramática linear $G$ esta na **forma normal linear forte** (FNLF) se está na FNL e os lados direitos das produções são da forma $aA$ ou $Aa$ para algum $a \in T \cup \{\lambda\}$ e $A \in V \cup \{\lambda\}$.

**Proposição 1.6** *Seja* $G$ *uma gramática linear. Existe uma gramática linear* $\widehat{G}$ *na FNLF tal que* $L(G) = L(\widehat{G})$.

**Exemplo 1.7** *A FNLF da gramática* $G'$ *do exemplo 1.5 é a gramática* $\widehat{G} = \langle\{S, A, B, C\}, \{a, b\}, S, \widehat{P}\rangle$ *onde* $\widehat{P}$ *é definido por*

$S \to aA$
$A \to Bb$
$B \to Sb \mid Cb \mid b$
$C \to Sb \mid b$

## Autômato linear não determinístico com $\lambda$-transições

Um **autômato linear não determinístico com $\lambda$-transições** ($\lambda$-ALN), consiste de dois conjuntos finitos disjuntos de estados ($Q$ e $P$) alguns dos quais serão considerados como estados de aceitação, uma fita de entrada infinita à direita e dividida em células, cada uma das quais comporta um símbolo de um alfabeto de entrada (ou é branco), duas cabeças leitoras e uma unidade de controle que administra a conduta do $\lambda$-ALN de acordo com a configuração corrente. A execução do $\lambda$-ALN começa com uma cadeia na fita de entrada, com a cabeça leitora esquerda apontando o símbolo mais à esquerda na fita e a cabeça leitora direita apontando o símbolo (diferente de branco) mais à direita na fita e tendo como estado corrente um estado especial de $Q$ chamado estado inicial. Um passo computacional num $\lambda$-ALN é feito como segue: a unidade de controle em função da classe que pertence o estado corrente (na primeira vez $Q$) usa a cabeça leitora esquerda ou direita para ler um símbolo da fita, logo move uma célula para a direita a cabeça leitora esquerda caso o estado corrente for de $Q$ ou uma célula para esquerda a cabeça leitora direita caso o estado corrente for de $P$. Ao mesmo tempo muda, seguindo um oráculo, para um estado desde um conjunto de possíveis estados. A unidade de controle de um $\lambda$-ALN também admite mudar de estado sem mover as cabeças leitoras, isto é sem ler símbolos da fita de entrada. A computação pára quando uma cabeça leitora cruza a outra, caso o estado corrente nesse momento for de aceitação o $\lambda$-ALN aceita a cadeia dada como entrada, caso contrário a rejeita. A figura 1 ilustra uma representação esquemática de um $\lambda$-ALN.

Formalmente, um $\lambda$-ALN é uma sêxtupla $M = \langle Q, P, \Sigma, \delta, q_0, F\rangle$ onde $Q$ e $P$ são conjuntos finitos disjuntos de estados, $\Sigma$ é um conjunto finito de símbolos de entrada (alfabeto), $q_0 \in Q$ é o estado de inicio, $F \subseteq (Q \cup P)$ é um conjunto de estados de aceitação e $\delta : (Q \cup P) \times (\Sigma \cup \{\lambda\}) \longrightarrow \wp(Q \cup P)$ é a função de transição.

Analogamente a autômatos finitos, cada $\lambda$-ALN tem associado um grafo dirigido chamado diagrama de transição. A única diferença com os diagramas de transição de autômatos finitos está na forma como distinguimos os estados de $Q$ com os estados de $P$. Para os primeiros usamos círculos enquanto para os outros usamos quadrados. Assim, um autômato finito não-determinístico pode ser visto como um $\lambda$-ALN onde $P = \emptyset$. Assim, $\lambda$-ALN é uma extensão natural para autômatos finitos $\lambda$-não determinísticos.

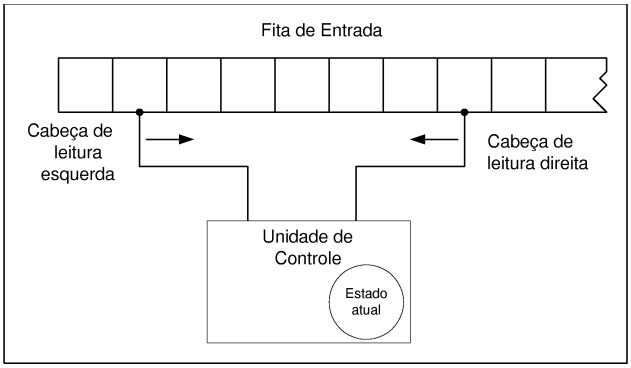


Figura 1: Representation esquemática de um $\lambda$-ALN.

**Exemplo 1.8** *A figura 2 ilustra o $\lambda$-ALN $M = \langle Q, P, \Sigma, \delta, q_0, F\rangle$ onde*

- $Q = \{q_0, q_1, q_2, q_3\}$
- $P = \{p_1, p_2, p_3, p_4\}$
- $\Sigma = \{a, b\}$
- $F = \{p_1, q_2\}$
- $\delta(q_0, a) = \{q_0, p_1\}$, $\delta(q_0, \lambda) = \{p_3\}$, $\delta(p_1, a) = \{p_2\}$, $\delta(p_2, a) = \{q_1\}$, $\delta(q_1, b) = \{p_1\}$, $\delta(p_3, b) = \{p_3, q_2\}$, $\delta(q_2, b) = \{q_3\}$, $\delta(q_3), b) = \{p_4\}$, $\delta(p_4, a) = \{q_2\}$ *e vazio para os demais (isto é, $\delta(q_0, b) = \emptyset$, $\delta(p_1, b) = \emptyset$, etc.).*

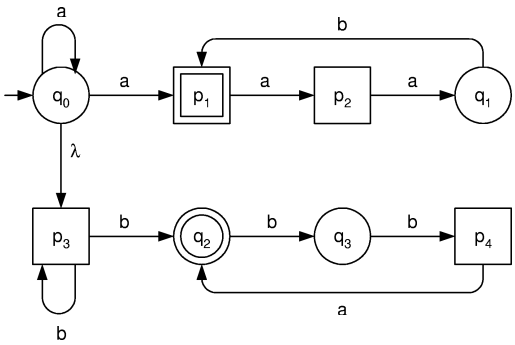


Figura 2: Diagrama de transição de um $\lambda$-ALN.

Uma **descrição instantânea** (DI) de um $\lambda$-ALN deve registrar o estado corrente, o que resta ler da cadeia de entrada e qual cabeça está ativa. Assim, uma DI é um par $(q, w) \in (Q \cup P) \times \Sigma^*$ significando que resta ainda ler $w$ na fita e que o estado atual é $q$. A cabeça leitora que está ativa é deduzida a partir do estado corrente.

O símbolo $\vdash$ denotará um movimento de uma DI a outra DI. Assim, para cada $q \in Q$, $q\prime \in Q \cup P$, $p \in P$, $w \in \Sigma^*$ e $a \in \Sigma \cup \{\lambda\}$

$(q, aw) \vdash (q\prime, w)$ é possível sss $q\prime \in \delta(q, a)$ e $(p, wa) \vdash (q\prime, w)$ é possível sss $q\prime \in \delta(p, a)$.

Usamos $\vdash^*$ para o fecho reflexivo e transitivo de $\vdash$, ou seja $\vdash^*$ representa uma quantidade arbitrária de movimentos. Assim, por exemplo, no exemplo 1.8, temos que

$$(q_0, aabbaaaa) \vdash^* (p_2, ba), \text{ pois}$$

$$\begin{aligned}(q_0, aabbaaaa) &\vdash (q_0, abbaaaa) \vdash (p_1, bbaaaa) \vdash (p_2, bbaaa)\\ &\vdash (q_1, bbaa) \vdash (p_1, baa) \vdash (p_2, ba)\end{aligned}$$

A linguagem aceita por um $\lambda$-ALN $M$ é o conjunto

$$L(M) = \{w \in \Sigma^* : (q_0, w) \vdash^* (q_f, \lambda) \text{ para algum } q_f \in F\}$$

Para o caso do $\lambda$-ALN $M$ no exemplo 1.8, $L(M) = L(G)$ em (1).

## $\lambda$-ALN's e linguagens lineares

Primeiro mostraremos que cada linguagem aceita por algum $\lambda$-ALN é linear.

**Teorema 1.9** *Seja $M = \langle Q, P, \Sigma, \delta, q_0, F\rangle$ um $\lambda$-ALN. Então existe uma gramática linear $G$ tal que $L(M) = L(G)$.*

**Demonstração:**Seja $G = \langle Q \cup P, \Sigma, q_0, P'\rangle$ onde

$$\begin{aligned}P' = \ & \{q_i \to aq_j : a \in \Sigma \cup \{\lambda\},\ q_i \in Q, q_j \in Q \cup P \text{ e } q_j \in \delta(q_i, a)\} \cup\\ & \{p_i \to q_j a : a \in \Sigma \cup \{\lambda\},\ p_i \in P, q_j \in Q \cup P \text{ e } q_j \in \delta(q_i, a)\} \cup\\ & \{q_f \to \lambda : q_f \in F\}\end{aligned}$$

Claramente, $L(M) = L(G)$. ■

Note que o $G$ resultante da prova está na FNLF.

**Exemplo 1.10** *Considere o $\lambda$-ALN na figura 2. A gramática linear resultante do algoritmo no teorema 1.9 é $G = \langle Q \cup P, \Sigma, q_0, P'\rangle$ onde $P'$ é composta pelas produções:*

$q_0 \to aq_0 \mid ap_1 \mid p_3$
$p_1 \to p_2 a \mid \lambda$
$p_2 \to q_1 a$
$q_1 \to bp_1$
$p_3 \to p_3 b \mid q_2 b$
$q_2 \to bq_3 \mid \lambda$
$q_3 \to bp_4$
$p_4 \to q_3 a$

Agora provaremos que cada linguagem linear é aceita por um $\lambda$-ALN.

**Teorema 1.11** *Seja $G = \langle V, T, S, \widehat{P}\rangle$ uma gramática linear. Então existe um $\lambda$-ALN $M$ tal que $L(G) = L(M)$.*

**Demonstração:**Sem perda de generalidade podemos assumir que $G$ está na FNLF. Seja $Q = \{A \in V : A \text{ é uma variável linear à direita}\} \cup \{C\}$, onde $C \notin V$ e $P = \{A \in V : A \text{ é uma variável linear à esquerda}\}$. Então

$$M = \langle Q, P, T, \delta, S, F\rangle,$$

onde $F = \{C\}$ e para cada $A \in Q \cup P$ e $a \in T \cup \{\lambda\}$, $\delta$ é definida por

$$\delta(A, a) = \begin{cases} \{B \in V : A \to aB \text{ ou } A \to Ba\} & \text{, se } A \to a \notin \widehat{P}\\ \{B \in V : A \to aB \text{ ou } A \to Ba\} \cup \{C\} & \text{, se } A \to a \in \widehat{P}\end{cases}$$

Claramente, $L(M) = L(G)$. ■

Em particular, para o exemplo 1.2, cuja FNLF está no exemplo 1.7, o $\lambda$-ALN obtido usando o algoritmo acima é ilustrado na figura 3. Identificando os estados do $\lambda$-ALN com as variáveis de $G$: $q_0 = S$, $p_3 = A$, $p_2 = B$, $p_4 = C$ e $q_2 \notin V$.

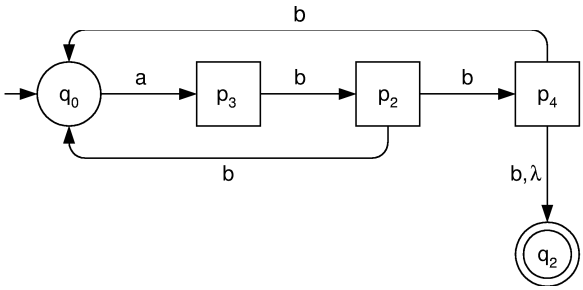


Figura 3: Diagrama de transição de um $\lambda$-ALN equivalente à gramática linear do exemplo 1.2.

Note que o algoritmo nos teoremas 1.9 e 1.11 são duais, no sentido que aplicando um e em seguida o outro sempre voltaremos ao original.

## Considerações Finais

Este trabalho introduz um modelo de autômatos intuitivo e simples para linguagens lineares que estende de maneira natural os autômatos finitos $\lambda$-não determinísticos. Mais ainda, os $\lambda$-ALN são intrínsecos no sentido que não consideram agentes externos ao modelo, como outros modelos de autômatos para esta classe de linguagens.

Os $\lambda$-movimentos nas modelos usuais de autômatos não são essenciais. Assim, por exemplo, $\lambda$-movimentos não aumentam o poder de aceitação de autômatos finitos (veja por exemplo [6]), de autômatos a pilha (veja, por exemplo, [4]) nem de máquinas de Turing (pela tese de Church-Turing). Embora não tenhamos provado que $\lambda$-movimentos não acrescentam poder computacional a $\lambda$-ALN, acreditamos fortemente nisso. Fica para trabalhos futuros provar esta conjectura.

## Referências

[1] Amar, V. and Putzolu, G.: On a Family of Linear Grammars. *Information and Control* **7**(1964):283–291.

[2] Autebert, J.M., Berstel, J. and Boasson, L.: Context-Free Languages and Pushdown Automata. In *Handbook of formal languages, vol. 1: word, language, grammar*, pages 111–174, Springer-Verlag, New York, 1997.

[3] Ginsburg, S.: *Algebraic and Automata-Theoretic Properties of Formal Languages*. North-Holland, Amsterdam, 1975.

[4] Harrison, M.A.: *Introduction to Formal Language Theory*, Addison-Wesley, Reading, MA, 1978.

[5] Hoogeboom, H.J. and Engelfriet, J.: Pushdown Automata. In: *Formal Languages and Applications*, Martin-Vide, C., Mitrana, V. and Paun G. (eds). Springer, Berlim, 2004.

[6] Hopcroft, J.E., Ullman, J.D.: *Introduction to Automata Theory, Languages and Computation*, Addison-Wesley, Reading, MA, 1979.

[7] Linz, P.: *An Introduction to Formal Language and Automata*. Jones and Bartlett Publisher, 2001.

[8] Mäkinen, E.: Inferring Finite Transducer. *Journal Brazilian Computational Society* **9**(1)(2003):5-8.

[9] Rosenberg, A.L.: A Machine Realization of the Linear Context-Free Languages. *Information and Control* **10**(1967): 175–188.

[10] Salomaa, A.: *Formal Languages*. Academic Press, 1973

[11] Takada, Y.: Grammatical Inference of Even Linear Languages based on Control Sets. *Information Processing Letter* **28**(4)(1988):193–199.