Tal como visto em AFD e AP, a linguagem L = $\{a^nb^n$ / $n > 0\}$ não consegue ser representada por um AFD, do mesmo modo que esta linguagem não consegue ser gerada por uma gramática regular.

Ao tentar desenvolver uma GR para a linguagem acima, percebemos que não consegue-se repetir a quantidade de b's, após os a's iniciais, fazendo com que a linguagem acima seja uma linguagem livre de contexto, passível de representação por uma Gramática Livre de Contexto.

Gramática Livre de Contexto

Uma linguagem livre de contexto é uma linguagem que pode ser descrito por uma Gramática Livre de Contexto (GLC). As linguagens livres de contexto apresentam um grau de complexidade maior, uma vez que são as linguagens que são reconhecidas por Autômatos com Pilha, utilizando a pilha como memória auxiliar, como vimos no Capítulo sobre AP.

As GLC são as gramáticas na qual todas as regras de produção pertencentes ao conjunto P são da forma:

- $A \rightarrow \beta$
- Onde $\beta \in (V \cup T)^*$
- $A \in V$

Antes de desenvolvermos a GLC para a gramática acima, vamos observar o desenvolvimento de GLC para linguagens regulares, ou seja, desenvolver GLC para as linguagens do Capítulo de GR.

A linguagem L = $\{a^nbbc^m$ / $n > 0$ e $m \geq 0\}$ representada em GLC pode ser montada observando a ideia de que a quantidade n de a seja representado por uma variável A, os dois b's sejam representados por um B e a quantidade m de c's seja representada por C. Assim construímos uma regra partindo do símbolo inicial S, na qual $S \rightarrow ABC$, e para cada uma das variáveis A, B e C representa os símbolos terminais da gramática, portanto a GLC para a linguagem acima pode ser representada por:

- GLC = (V = {S, A, B, C}, T = {a, b, c}, P, S = S), e P possui as seguintes produções:
  - 1) $S \rightarrow ABC$
  - 2) $A \rightarrow a$
  - 3) $A \rightarrow aA$
  - 4) $B \rightarrow bb$
  - 5) $C \rightarrow cC$

- 6) C → λ

Como nas GLC podemos ter mais de uma variável do lado direito das regras de produção, a construção das gramáticas como uma GLC fica mais fácil do que uma GR.

Para a linguagem L = {$a^n b^n$ / n > 0} e linguagens nas quais temos definido uma ordem (todos os a's primeiro, depois todos os b's), devemos pensar na GLC de modo que ao gerar um símbolo inicial de uma palavra, gera-se um símbolo no final da palavra. De modo análogo as GR, que temos o crescimento da palavra para direita nas GLD e GLUD, e o crescimento da palavra para a esquerda nas GLE e GLUE, nas GLC podemos ter o crescimento das palavras para o centro, partindo do símbolo inicial para a metade da palavra e partindo do último símbolo para a metade da palavra. Isto é feito utilizando-se de regras de produção que gerem sempre uma mesma quantidade de a e b juntos, por exemplo, S → aSb, que a qualquer chamada efetuada por essa regra de produção gera sempre um a e um b, garantindo que independente da quantidade de vezes chamada a regra de produção, a quantidade de a e b será igual.

A GLC para L = {$a^n b^n$ / n > 0} pode ser vista abaixo:

- GLC = (V = {S, A}, T = {a, b}, P, S = S), e P possui as seguintes produções:
    - 1) S → aAb
    - 2) A → aAb
    - 3) A → λ

Para a menor palavra desta linguagem (ab), temos a seguinte derivação:

Símbolo inicial	Regra utilizada
S	1) S → aAb
a**A**b	3) A → λ
aλb	aλb = ab
ab	ab é uma Palavra da Linguagem

Para a palavra aaabbb temos a seguinte derivação

Símbolo inicial	Regra utilizada
S	1) S → aAb
a**A**b	2) A → aAb
aa**A**bb	2) A → aAb
aaa**A**bbb	3) A → λ
aaa**λ**bbb	aaaλbbb = aaabbb

aaabbb	aaabbb é uma Palavra da Linguagem

Observando a derivação acima percebemos que a primeira derivação utilizando a regra 1, gera a sentença aAb, na qual este a será o primeiro símbolo da palavra e este b será o último da palavra. A segunda derivação utilizando a regra 2, gera como sentença a palavra aaAbb, gerando o segundo a da palavra e o penúltimo b da palavra e assim por diante. Deste modo podemos observar que as derivações "crescem" para o centro da palavra, fazendo com que a primeira metade da palavra contenha uma quantidade n de a's, seguido de uma quantidade n de b's com n maior que 0, que é a representação da linguagem acima.

Para a linguagem $L = \{a^n b^m c^n \ / \ n > 0, m > 0\}$ temos a seguinte GLC

- GLC = (V = {S, A, B}, T = {a, b, c}, P, S = S), e P possui as seguintes produções:
    - 1) $S \rightarrow aAc$
    - 2) $A \rightarrow aAc$
    - 3) $A \rightarrow bB$
    - 4) $B \rightarrow bB$
    - 5) $B \rightarrow \lambda$

O desenvolvimento da gramática necessita primeiro colocar os símbolos a e c, no início e no fim da palavra respectivamente, e após inserir estes símbolos acrescentar a quantidade m de b entre eles. Isto é feito utilizando-se da regra $S \rightarrow aAc$, que garante a obrigatoriedade de passar por esta transição, inserindo pelo menos um a e um c na palavra ($n > 0$). Após a chamada desta regra de produção, podemos usar a regra 2 para aumentar a quantidade de a e c na palavra, ou podemos usar a regra 3 e inserir a menor quantidade possível de b na palavra ($m > 0$) e entre o a e c. Feito esta chamada a regra 3, a derivação continua gerando mais símbolos b caso necessário pela regra 4, ou terminando gerando a palavra vazia pela regra 5.

A derivação de uma palavra para a linguagem acima pode ser vista na tabela abaixo

Símbolo inicial	Regra utilizada
S	1) $S \rightarrow aAc$
a**A**c	2) $A \rightarrow aAc$
aa**A**cc	2) $A \rightarrow aAc$
aaa**A**ccc	3) $A \rightarrow bB$
aaa**bB**ccc	4) $B \rightarrow bB$
aaab**bB**ccc	5) $B \rightarrow \lambda$
aaabbλccc	aaabbλccc = aaabbccc
aaabbccc	aaabbccc é uma Palavra da Linguagem

Outro exemplo para a linguagem L = $\{a^ib^jc^k$ / $k = i + j$, $i \geq 0$, $j \geq 0\}$ pode ser construído como uma GLC. Primeiramente devemos analisar a expressão da linguagem, que indica que a quantidade de c da palavra, eh igual a quantidade de a somada com a quantidade de b. De outro modo podemos pensar que para cada a temos um c na palavra, assim como para cada b também temos um c, sempre respeitando a ordem dos símbolos que indica que todos os a's vem primeiro, depois todos os b's e por fim todos os c's. Esta analise na linguagem pode ser representada com uma expressão diferente (L = $\{a^nb^mc^mc^n$ / $n \geq 0$, $m \geq 0\}$) para a linguagem, porém aceitando as mesmas palavras da linguagem acima, uma vez que a quantidade de c (n + m) ainda é igual a quantidade de a somado com a quantidade de b (n + m), sendo que ambos a e b não são obrigados a existir nas palavras da linguagem (maior ou igual a zero). A gramática para tal linguagem pode ser visualizada abaixo

- GLC = (V = {S, A}, T = {a, b, c}, P, S = S), e P possui as seguintes produções:
    - 1) S → aSc
    - 2) S → bAc
    - 3) S → λ
    - 4) A → bAc
    - 5) A → λ

Para a menor palavra da linguagem (palavra vazia), a derivação começa com a regra de produção 3. Para palavras que não contenham o símbolo b, a derivação começa com a regra de produção 1, é derivada quantas vezes for necessária, e finaliza com a regra 3. Para palavras que não contenham o símbolo a, a derivação começa com a regra de produção 2, é derivada quantas vezes for necessária usando a regra de produção 4 e finaliza com a regra 5. E por fim, para palavras com a e b, a derivação inicia-se com a regra de produção 1, passa pela regra de produção 2, e finaliza com a regra 5. Exemplos destas derivações podem ser vistos na tabela abaixo.

Símbolo inicial	Regra utilizada
S	1) S → aSc
a**S**c	1) S → aSc
aa**S**cc	1) S → aSc
aaa**S**ccc	3) S → λ
aaaλccc	aaaλccc = aaaccc
aaaccc	aaaccc é uma Palavra da Linguagem

Símbolo inicial	Regra utilizada
S	2) S → bAc

b**A**c	4) A → bAc
bb**A**cc	4) A → bAc
bbb**A**ccc	5) A → λ
bbbλccc	bbbλccc = bbbccc
bbbccc	bbbccc é uma Palavra da Linguagem

Símbolo inicial	Regra utilizada
S	1) S → aSc
a**S**c	1) S → aSc
aa**S**cc	2) S → bAc
aab**A**ccc	4) A → bAc
aabb**A**cccc	4) A → bAc
aabbb**A**ccccc	5) A → λ
aabbb**λ**ccccc	aabbbλccccc = aabbbccccc
aabbbccccc	aabbbccccc é uma Palavra da Linguagem

A linguagem L = $\{w \in \{a,b\}^* \mid |w|_a = |w|_b\}$ pode ser descrita como a linguagem na qual a quantidade de a's é igual a quantidade de b's em qualquer ordem, sendo que algumas palavras que compõem a linguagem são L = {λ, ab, ba, aabb, abab, abba, bbaa, baba, baab, ...}. A GLC para esta linguagem pode ser vista abaixo

- GLC = (V = {S, A, B}, T = {a, b}, P, S = S), e P possui as seguintes produções:
  - 1) S → aB
  - 2) S → bA
  - 3) S → λ
  - 4) A → aS
  - 5) A → bAA
  - 6) B → bS
  - 7) B → aBB

A derivação de palavras na GLC acima segue a seguinte ideia: começamos a derivação pelas regras de produção 1, 2 ou três. Com a regra 3 temos a palavra vazia que é a menor palavra da linguagem. Quando utilizamos a regra 1, indica que inserimos um a na palavra, seguido da variável B. Este B se utilizado na regra 6, gera um b na palavra seguido da variável S, que pode ser utilizada nas regras 1, 2 e 3 novamente. Ainda quando utilizamos na primeira derivação a regra 1, porém na segunda derivação usamos a regra 7, é o caso no qual inserimos um a na primeira derivação, na segunda derivação outro a. Isto indica que deve-se ter dois b’s para que a quantidade de B seja igual a quantidade de a. Estes dois b’s são gerados pelos dois símbolos B gerados após a derivação da regra 7. Exemplos de derivações para esta linguagem podem ser visualizados nas tabelas abaixo.

Símbolo inicial	Regra utilizada
S	1) S → aB
a**B**	7) B → aBB
aa**B**B	6) B → bS
aab**S**B	3) S → λ
aab**B**	6) B → bS
aabb**S**	3) S → λ
aabbλ	aabbλ = aabb
aabb	aabb é uma Palavra da Linguagem

Símbolo inicial	Regra utilizada
S	2) S → bA
b**A**	4) A → aS
ba**S**	1) S → aB
baa**B**	7) B → aBB
baaa**B**B	6) B → bS
baaab**S**B	3) S → λ
baaabλ**B**	6) B → bS
baaabb**S**	3) S → λ
baaabbλ	baaabbλ = baaabb
baaabb	baaabb é uma Palavra da Linguagem

Com as Gramática Livres de Contexto podemos montar uma linguagem na qual aceita-se parênteses balanceados, operadores matemáticos e um operando, esta gramática pode ser visualizada abaixo

- GLC = (V = {S}, T = {(,), *, +, x}, P, S = S), e P possui as seguintes produções:
    - 1) S → S + S
    - 2) S → S * S
    - 3) S → (S)
    - 4) S → x

Esta gramática pode ser descrita como uma gramática ambígua, uma vez que pode-se gerar derivações diferentes para a mesma palavra.

Gramática Sensível ao Contexto

Uma linguagem sensível ao contexto é uma linguagem que pode ser descrito por uma gramática sensível ao contexto, e esta gramática é GSC quando todas as regras de produção pertencentes ao conjunto P são da forma:

- $\alpha \rightarrow \beta$
- Tal que $| \alpha | \leq | \beta |$ exceto quando $\beta = \lambda$
- Onde: $\beta \in (V \cup T)^*$ e $\alpha \in (V \cup T)^+$

Um exemplo de gramática sensível ao contexto pode ser visualizada abaixo

- G = (V = {S, B, C}, T = {a, b, c}, P, S = S), onde P:
    - $S \rightarrow aSBC$
    - $S \rightarrow aBC$
    - $CB \rightarrow BC$
    - $aB \rightarrow ab$
    - $bB \rightarrow bb$
    - $bC \rightarrow bc$
    - $cC \rightarrow cc$

Esta GSC gera a seguinte linguagem $L = \{a^n b^n c^n \mid n > 0\}$ e as derivações de palavras para esta linguagem ficam de exercício para o leitor

Gramática Irrestrita

Uma linguagem enumerável recursivamente é uma linguagem que pode ser descrito por uma gramática irrestrita. A GI é uma gramática na qual todas as regras de produção pertencentes ao conjunto P são da forma:

- $\alpha \rightarrow \beta$
- Onde: $\beta \in (V \cup T)^*$ e $\alpha \in (V \cup T)^+$

Para este tipo de gramática, um exemplo pode ser visualizado abaixo, para a mesma linguagem $L = \{a^n b^n c^n \mid n > 0\}$

- G = (V = {S, C}, T = {a, b, c}, P, S = S), onde P:
    - $S \rightarrow abc$

- $S \rightarrow \lambda$
- $ab \rightarrow aabbC$
- $Cb \rightarrow bC$
- $Cc \rightarrow cc$

Bibliografia

VIEIRA, NEWTON JOSÉ. **Introdução aos Fundamentos da Computação**. São Paulo. Pioneira Thomson Learning. 2006;
MENEZES, P. B.; DIVERIO, T. A.; **Teoria da Computação: Máquinas Universais e Computabilidade**; 3ª edição Bookman 2011
MENEZES, P. B. ; **Linguagens Formais e Autômatos**. 6ª edição. Ed. Artmed. 2011
SIPSER M. **Introdução à Teoria da Computação**. 2 ed. Cengage Learning.2007
MORET, B. M. **"Theory of Computation".** Addison-Wesley, 1998.
HOPCROFT, JOHN E.; ULLMAN, JEFFREY D.; MOTWANI, RAJEEV **Introdução à Teoria de Autômatos, Linguagens e Computação** Ed.Campus 2002
SILVA, FLAVIO SOARES CORRÊA; MELO, ANA CRISTINA VIEIRA; **Modelos Clássicos de Computação** Ed. Thomson 1ª Edição 2006