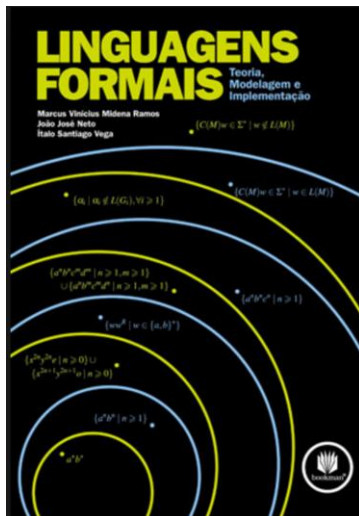


TEORIA: GRAMÁTICAS E HIERARQUIA DE CHOMSKY



Nossos **objetivos** nesta aula são:

- conhecer o conceito de gramática e a Hierarquia de Linguagens de Chomsky
- praticar com gramáticas



Para esta semana, usamos como referências as **Seções 2.3 (Gramáticas)** e **2.6 (Hierarquia de Chomsky)** do nosso livro da referência básica:

RAMOS, M.V.M., JOSÉ NETO, J., VEJA, I.S. **Linguagens Formais: Teoria, Modelagem e Implementação**. Porto Alegre: Bookman, 2009.

Não deixem de ler estas seções depois desta aula!

TEORIA: GRAMÁTICAS E HIERARQUIA DE CHOMSKY

- Uma **gramática** é uma quádrupla $G=(N,T,P,S)$, onde:
 - **N**: conjunto finito e não-vazio de elementos chamados **não-terminais**
 - **T**: conjunto finito e não-vazio de elementos chamados **terminais**
 - **P**: conjunto finito e não-vazio de elementos chamados **produções** (ou **regras**)
 - $S \in N$ é a **raiz** ou **símbolo inicial** da gramática.
- Ex: $G = (\{S\}, \{a,b\}, \{S \rightarrow ab, S \rightarrow aSb\}, S)$
- Uma **derivação** em uma gramática G é uma sequência de substituições obtidas a partir das produções de G , tendo como base o seu símbolo inicial.
 - Ex: $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaabbb$

- A linguagem **L(G)** gerada por uma gramática G é dada pelo conjunto:

$$L(G) = \{ \omega \in T^* \mid S \Rightarrow^+ \omega \}$$

- Ex: $G = (\{S\}, \{a,b\}, \{S \rightarrow ab, S \rightarrow aSb\}, S)$ $L(G) = \{a^n b^n \mid n \geq 1\}$
- **Hierarquia de Chomsky** : classificação das linguagens formais, realizada em função do tipo de produção $\alpha \rightarrow \beta$ da gramática associada à linguagem. Para cada tipo de linguagem, temos um reconhecedor específico:

Linguagem	Gramática	Tipo de Produção	Reconhecedor
Regular	Regular (ou Tipo 3)	<ul style="list-style-type: none"> ▪ α não-terminal ▪ β terminal ou β não-terminal ou $\beta = NT$ ou $\beta = TN$ ou $\beta = \epsilon$ 	Autômato Finito
Livre de Contexto	Livre de Contexto (GLC) (ou Tipo 2)	<ul style="list-style-type: none"> ▪ α não-terminal ▪ $\beta = (N \cup T)^*$ 	Autômato à Pilha
Sensível ao Contexto	Sensível ao Contexto (ou Tipo 1)	<ul style="list-style-type: none"> ▪ $\alpha = (N \cup T)^* N (N \cup T)^*$ ▪ $\beta = (N \cup T)^*$ ▪ $\alpha \leq \beta$ 	Autômato Linearmente Limitado (Máquina de Turing com Fita Limitada)
Recursivamente enumerável	Irrestrita (ou Tipo 0)	<ul style="list-style-type: none"> ▪ $\alpha = (N \cup T)^* N (N \cup T)^*$ ▪ $\beta = (N \cup T)^*$ 	Máquina de Turing

EXERCÍCIO TUTORIADO

Considere-se a gramática $G=(N,T,P,S)$ definida como:

$N = \{ \text{expr} \}$
 $T = \{ \text{num}, + \}$
 $P = \{ \text{expr} \rightarrow \text{expr} + \text{expr}, \text{expr} \rightarrow \text{num} \}$
 $S = \text{expr}$

- Mostre que a palavra 345+60+20 pode ser derivada por esta gramática.

- Descreva, informalmente, qual a linguagem formal gerada por esta gramática.

EXERCÍCIO COM DISCUSSÃO PAREADA

Considere-se a gramática $G=(N,T,P,S)$ definida como:

$N = \{ \text{PAR} \}$

$T = \{ (,) \}$

$P = \{ \text{PAR} \rightarrow (), \text{PAR} \rightarrow (\text{PAR}), \text{PAR} \rightarrow \text{PAR PAR} \}$

$S = \text{PAR}$

- Mostre que a palavra $(())()$ pode ser derivada por esta gramática.

- Descreva, informalmente, qual a linguagem formal gerada por esta gramática.

PROBLEMA

Uma construção bastante comum em linguagens imperativas como C e Pascal é o comando de atribuição com expressões aritméticas, conforme mostrado no exemplo (em C) abaixo:

```
valor = (2+3*(60/884));
```

Para que se possa construir um compilador C que consiga verificar se uma atribuição possui algum erro sintático, é necessária uma gramática que consiga estabelecer as regras sintáticas de tal construção.

Construa uma gramática livre de contexto (GLC) que permita reconhecer atribuições com expressões aritméticas na linguagem C.

EXERCÍCIOS EXTRA-CLASSE

1. Classifique as gramáticas $G=(N,T,P,S)$ abaixo em regular, livre de contexto, sensível ao contexto ou irrestrita:

(a) $N = \{ S \}$

$T = \{ a \}$

$P = \{ S \rightarrow a, S \rightarrow aS \}$

S

(b) $N = \{ \text{expr}, \text{termo} \}$

$T = \{ \text{num}, +, *, (,) \}$

$P = \{ \text{expr} \rightarrow \text{termo} + \text{termo}, \text{expr} \rightarrow \text{termo}, \text{termo} \rightarrow \text{num}, \text{termo} \rightarrow (\text{expr})^*(\text{expr}) \}$

$S = \text{expr}$

(c) $N = \{ S, A, B \}$

$T = \{ a, b \}$

$P = \{ S \rightarrow AB, AB \rightarrow BA, A \rightarrow aA, B \rightarrow Bb \}$

S

(d) $N = \{ S, A, B, C \}$

$T = \{ a, b, c \}$

$P = \{ S \rightarrow AB, AB \rightarrow BAC, A \rightarrow aA, BAC \rightarrow Bb, C \rightarrow cCc \}$

S

2. Considere-se o alfabeto $\Sigma = \{0,1\}$. Construa uma gramática que consiga gerar a linguagem dos números binários que são potência de 2.

3. Considere-se a gramática $G=(N,T,P,S)$ definida como:

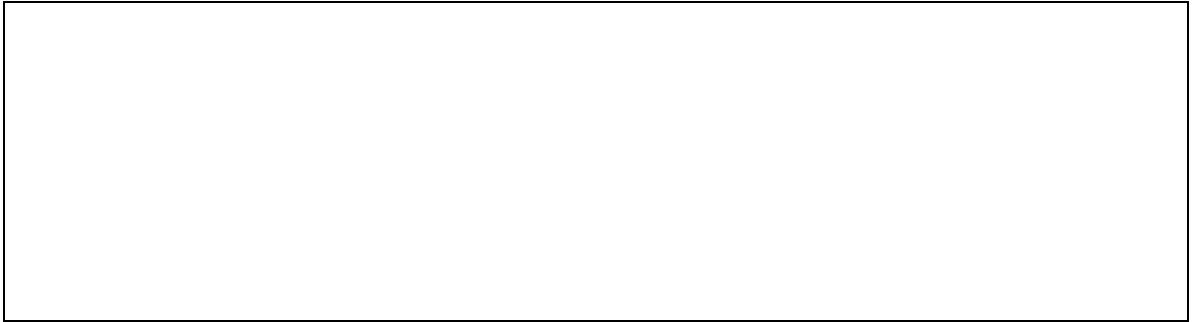
$N = \{ \text{num} \}$

$T = \{ +, -, \text{digitos}, \text{digito} \}$ // digito é qualquer símbolo dentre 0,1,2,3,4,5,6,7,8,9

$P = \{ \text{num} \rightarrow + \text{digitos}, \text{num} \rightarrow - \text{digitos}, \text{num} \rightarrow \text{digitos}, \text{digitos} \rightarrow \text{digito}, \text{digitos} \rightarrow \text{digito digitos} \}$

$S = \text{num}$

Descreva a linguagem gerada por esta gramática:



4. Por que o diagrama de inclusões de linguagens na Hierarquia de Chomsky, mostrado abaixo, é válido ?

