Prof José Rui

1 – Introdução e Conceitos básicos

Aula 02

Sumário

Capítulo 1 – Introdução e Conceitos básicos

Sumário

Introdução

- Linguagem
  - Dicionário: o uso da palavra articulada ou escrita como meio de expressão e comunicação entre pessoas
  - Não é suficiente preciso para definir modelos matemáticos
  - Então faremos algumas definições formais para nosso estudo
- Para definir linguagem formal
  - Precisamos de um alfabeto
  - E um cadeia de caracteres, ou palavras

Alfabeto

- DEFINIÇÃO: Conjunto **finito** de símbolo ou caractere
- Notação: Σ
  - Simbolos ou caracteres:
    - Entidade abstrata básica
    - Ex: letra, digitos...
  - São alfabetos
    - {a, b, c}
    - Ø, conjunto vazio
  - Não são alfabetos
    - **N**, conjunto dos naturais
    - {a, aab, bbb, aba, ...}

Alfabeto

- DEFINIÇÃO: Conjunto **finito** de símbolo ou caractere
- Alfabeto em Linguagem de programação
  - Conjunto de todos os simbolos utilizados na linguagem
    - Letras
    - Digitos
    - Caracteres especiais “>, <, <=, /, *” etc
    - Espaços ou brancos
- Alfabeto binário
  - Domínio de valores de um bit
  - Podemos usar {a,b} ou {0,1}
  - Muito usado para simplificar as abordagens estudas

Palavra

- Também chamada de:
  - Cadeia de caracteres
  - Sentença
- DEFINIÇÃO: **Sequência finita** de símbolos justapostos ou caractere
- Notação: w
- São palavras no alfabeto {a,b}
    - ab, bb, aaa
    - ε, cadeia vazia
  - Não são palavras no alfabeto{a,b}
    - ab..., aaa..., abc

Palavra

- Elementos de uma palavra
  - Prefixo
    - Qualquer sequência inicial de simbolos de uma palavra
  - Sufixo
    - Qualquer sequência final de simbolos de uma palavra
  - Subpalavra
    - Qualquer sequência de simbolos contiguos de uma palavra
  - Comprimento |w|
    - Número de caracteres de uma palavra
  - Ex:

    w = abcb palavra sobre o alfabeto {a,b,c}
    - ε, a, ab, abc, abcb são prefixos
    - ε, a, ab, abc, abcb são sufixos
    - Todos sufixo e prefixo é uma subpalavra
    - |w| = 4
    - |ε| = 0

Palavra

- Exemplo de palavra em Linguagem de programação
  - Um Bloco de programa
  - Uma palavra-chave, if, while...
  - Um programa inteiro

Palavra

- Concatenação
  - Operação binária sobre um conjunto de palavras
  - Associa duas palavras
  - É a justaposição da primeira com a segunda
- Propriedades
  - Elemento neutro: $\varepsilon w = w = w \varepsilon$
  - Associativa: $v (w t) = (v w) t$
- Exemplos

  $\Sigma = \{a,b\}$ um alfabeto. Para v = baaab, w=bb
  - vw = baaabbb
  - $v\varepsilon$ = baaab

Palavra

- Concatenação sucessiva
  - b^4 = bbbb
  - $a^0 = \varepsilon$
  - a^3b^5 = aaabbbbb
- Se Σ é um alfabeto
  - **Σ*** é conjunto de todas as palavras possíveis sobre Σ
  - **$\Sigma^+ = \Sigma^* - \varepsilon$**

Sumário

Linguagem Formal

- DEFINIÇÃO: é um subconjunto de **Σ***
  - Isto é, um subconjunto de todas “palavras” possíveis dentro de um alfabeto
- Notação: L

Linguagem Formal

- DEFINIÇÃO: é um subconjunto de **Σ***
  - Isto é, um subconjunto de todas “palavras” possíveis dentro de um alfabeto
- Notação: L
- Exemplos
  - Ø e {ε} são linguagens sobre qualquer alfabeto
  - Σ^* e Σ^+ são linguagens sobre qualquer alfabeto
  - Conjunto de palindromes sobre $\Sigma = \{a, b\}$
    - {ε, a, b, aa, aba, bab, aabbaa, …}

Linguagem Formal

- DEFINIÇÃO: é um subconjunto de **Σ^***
  - Isto é, um subconjunto de todas "palavras" possíveis dentro de um alfabeto
- Notação: L
- Exemplos
  - Ø e {ε} são linguagens sobre qualquer alfabeto
  - Σ^* e Σ^+ são linguagens sobre qualquer alfabeto
  - Conjunto de palindromes sobre $\Sigma = \{a, b\}$
    - {ε, a, b, aa, aba, bab, aabbaa, …}
  - Linguagem de programação
    - Conjunto de todos programas (palavras) da linguagem
    - Conjunto de palavras chave de uma linguagem, if, while, do, int, integer...

Linguagem Formal

- Linguagem de programação
  - C++
    - {for, if, while, do, int, new, …}
  - Delphi
    - {for, if, while, do, integer, begin, end …}
  - Java
    - {for, if, while, do, int, forech, …}

- Como podemos notar
- Cada linguagem tem um conjunto de palavras que aceita
- Ou seja, esse conjunto de palavras aceitas que chamamos de linguagem

Linguagem Formal

- Linguagem de programação
  - A palavra “begin” é aceita pela linguagem c++ ?
  - A palavra “how” é aceita pela linguagem português ?
  - A palavra “ teste := 2*x;” é aceita pela linguagem Java?

Linguagem Formal

- DEFINIÇÃO: é um subconjunto de **Σ***
  - Isto é, um subconjunto de todas “palavras” possíveis dentro de um alfabeto
- Notação: L
- Exemplos
  - Ø e {ε} são linguagens sobre qualquer alfabeto
  - Σ^* e Σ^+ são linguagens sobre qualquer alfabeto
  - Conjunto de palindromes sobre Σ = {a, b}
    - {ε, a, b, aa, aba, bab, aabbaa, …}
  - Linguagem de programação c++
    - Conjunto de todos os programas (palavras) da linguagem
    - Conjunto de palavras: if, while, new, int, double...
    - Atenção, um programa como um todo é considerado uma “palavra”

Sumário

Gramática

- DEFINIÇÃO:
  - Conjunto **finito de regras**
  - Quando aplicada sucessivas vezes, geram palavras
- O conjunto de todas as palavras geradas por uma gramática
  - Define a Linguagem
- A gramática também é usada para definir semântica

Gramática

- DEFINIÇÃO FORMAL:
  - Gramatica de Chomsky, gramática irrestrita, ou apenas gramática

$$G = (V, T, P, S)$$

  - V .... conjunto finito de símbolos, variáveis ou não-terminais
  - T .... conjunto finito de símbolos, terminais
  - P .... produções
    - $(V \cup T)^+ \rightarrow (V \cup T)^+$, relação finita
    - Par de relação, regra de produção ou produção
  - S .... elemento diferente de V, variável inicial

Gramática

- Representação de uma regra de produção

  $\alpha \rightarrow \beta$

  - Se α tem mais de uma produção

    $\alpha \rightarrow \beta_1$

    $\alpha \rightarrow \beta_2$

    $\alpha \rightarrow \beta_3$

  - Pode-se abreviar para

    $\alpha \rightarrow \beta_1 \mid \beta_2 \mid \beta_3$

Gramática

- Derivação
  - É o processo de aplicar as regras de produções
  - Inicia-se sempre de **S**
  - Permite gerar as palavras da linguagem

- Na prática derivar é:
  - Substituir uma uma **subpalavra**, segunda uma regra de produção
  - Ex:
    - Para estas regra de produções, $\alpha \rightarrow \beta_1 | \beta_2 | \beta_3$
    - Podemos substituir α por β_1 , por β_2 , ou por β_3

Prof José Rui

Gramática

- Exemplo:

$$G = (V, T, P, S)$$

  - V = {N, D}
  - T = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
  - P = {

    $N \rightarrow D$

    $N \rightarrow DN$

    $D \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

    }
  - S = {N}

Gramática

- Exemplo:

$$G = (V, T, P, S)$$

  - V = {N, D}
  - T = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
  - P = {

    $N \rightarrow D$

    $N \rightarrow DN$

    $D \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

    }
  - S = {N}

Para facilitar vamos numerar as produções

Gramática

- Exemplo:

$$G = (V, T, P, S)$$

- V = {N, D}
- T = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
- P = {

  1 N → D

  2 N → DN

  3 D → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

  }
- S = {N}

Para facilitar vamos numerar as produções

Gramática

- Exemplo:
  - Vamos derivar o numero 243

P = {
  N → D 1
  N → DN 2
  D → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 3
}

N →

Aplicando 2 gera:

Gramática

- Exemplo:
  - Vamos derivar o numero 243

P = {
N → D 1
N → DN 2
D → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 3
}

N → Aplicando 2 gera:

DN→

Gramática

- Exemplo:
  - Vamos derivar o numero 243

P = {
N → D 1
N → DN 2
D → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 3
}

N →

DN→ Aplicando 3 gera:

Gramática

- Exemplo:
  - Vamos derivar o numero 243

P = {
N → D 1
N → DN 2
D → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 3
}

N →

DN → Aplicando 3 gera:

2N →

Prof José Rui

Gramática

- Exemplo:
  - Vamos derivar o numero 243

P = {
  N → D   1
  N → DN   2
  D → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9   3
}

N →

DN →

2N →   Aplicando 2 gera:

Gramática

- Exemplo:
  - Vamos derivar o numero 243

P = {
    N → D (1)
    N → DN (2)
    D → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 (3)
}

N →

DN →

2N → Aplicando 2 gera:

2DN →

Gramática

- Exemplo:
  - Vamos derivar o numero 243

P = {
  N → D 1
  N → DN 2
  D → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 3
}

N →

DN →

2N →

2DN →   Aplicando 3 gera:

Gramática

- Exemplo:
  - Vamos derivar o numero 243

P = {
N → D 1
N → DN 2
D → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 3
}

N →

DN →

2N →

2DN → Aplicando 3 gera:

24N →

Gramática

- Exemplo:
  - Vamos derivar o numero 243

P = {
N → D 1
N → DN 2
D → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 3
}

N →

DN →

2N →

2DN →

24N → Aplicando 1 gera:

Gramática

- Exemplo:
  - Vamos derivar o numero 243

P = {
N → D   1
N → DN   2
D → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9   3
}

N →

DN →

2N →

2DN →

24N →   Aplicando 1 gera:

24D →

Gramática

- Exemplo:
  - Vamos derivar o numero 243

P = {
N → D 1
N → DN 2
D → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 3
}

N →

DN →

2N →

2DN →

24N →

24D → Aplicando 3 gera:

Gramática

- Exemplo:
  - Vamos derivar o numero 243

P = {
N → D 1
N → DN 2
D → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 3
}

N →

DN →

2N →

2DN →

24N →

24D →

243

Gramática

- Exercicio:

$$G = (V, T, P, S)$$

  - $V = \{N, D\}$
  - $T = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
  - $P = \{$

    $N \rightarrow D$

    $N \rightarrow DN$

    $D \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

    $\}$
  - $S = \{N\}$
- Faça as seguintes derivações
  - 1000
  - 02
  - 5890
  - 3,45

Gramática

- Podemos concluir que:
  - A gramática anterior produz a linguagem dos numeros naturais
  - Denotamos **LINGUAGEM GERADA** por:
    - **G: L(G)** ou **GERA(G)**
  - Para que duas linguagens G_1 e G_2 sejam iguais ou equivalentes

$$\mathbf{L(G_1) = L(G_2)}$$

  - Isto é, o conjunto das palavras aceitas devem ser iguais

Gramática

- Convencionaremos que:
  - A, B, C, S, …, T, letras maiúsculas para Variáveis
  - a, b, c, s, …, t, letras minúsculas para terminais
  - u, v, w, x, y, z, para palavras de simbolos terminais

Gramática

- Exemplo 02:
  - G = (V, T, P, S)
  - V = {S, X, Y, A, B, F}
  - T = {a,b}
  - P = {

    S → XY

    X → XaA | XbB | F

    Aa → aA

    Ab → bA

    AY → Ya

    Ba → aB

    Bb → bB

    BY → Yb

    Fa → aF

    Fb → bF

    FY → ε

    }
  - S

- Esta gramática gera a linguagem:

  {ww | w é palavra de {a,b}* }

- Isto é:
  - ab<u>ab</u>
  - abb<u>abb</u>
  - aaa<u>aaa</u>
- Exercicio, derive:
- ba<u>ba</u>
- bbba<u>bbba</u>

Prof José Rui

Gramática

- Exemplo 03:
  - G = (V, T, P, S)
  - V = {A, B, C}
  - T = {a,b}
  - P = {

    S → aA | bB

    A → bB | aC

    B → aA | bC

    C → a | b | aC | bC | ε

    }
  - S

- Esta gramática gera a linguagem:

  ?

- Testar se as palavras são aceitas
  - abab
  - aab
  - abaab
  - abb
  - babbabab

Gramática

- Exemplo 03:
  - G = (V, T, P, S)
  - V = {A, B, C}
  - T = {a,b}
  - P = {

    S → aA | bB

    A → bB | aC

    B → aA | bC

    C → a | b | aC | bC | ε

    }
  - S

- Esta gramática gera a linguagem:

  {w | w tem pelo menos ***aa*** ou ***bb*** como subpalavra}

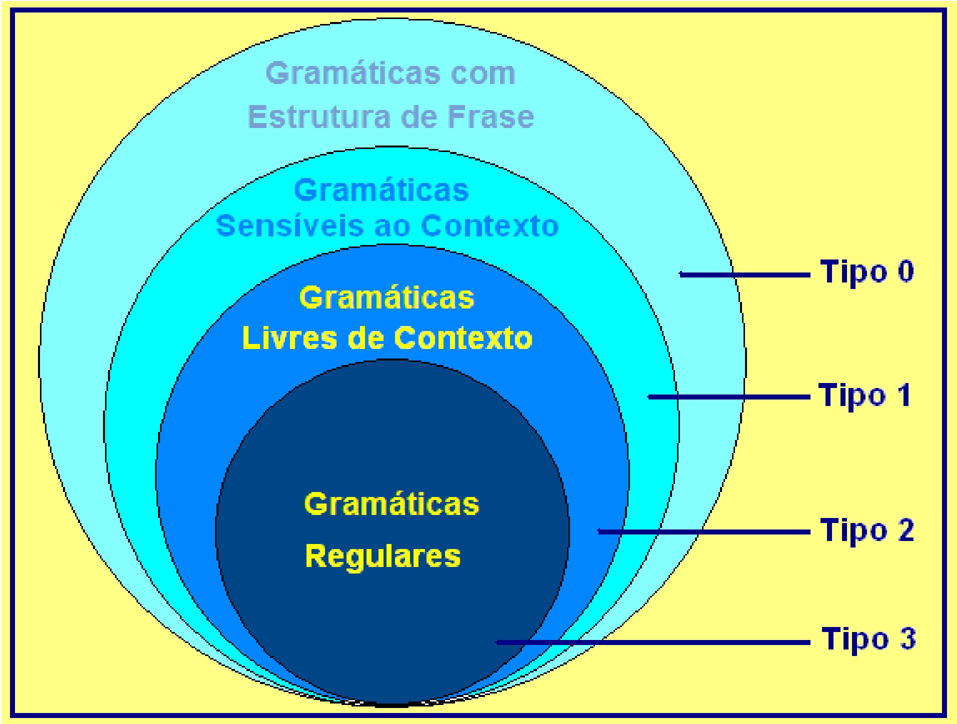
- Testar se as palavras são aceitas
  - abab
  - aab
  - abaab
  - abb
  - babbabab

Prof José Rui

Sumário

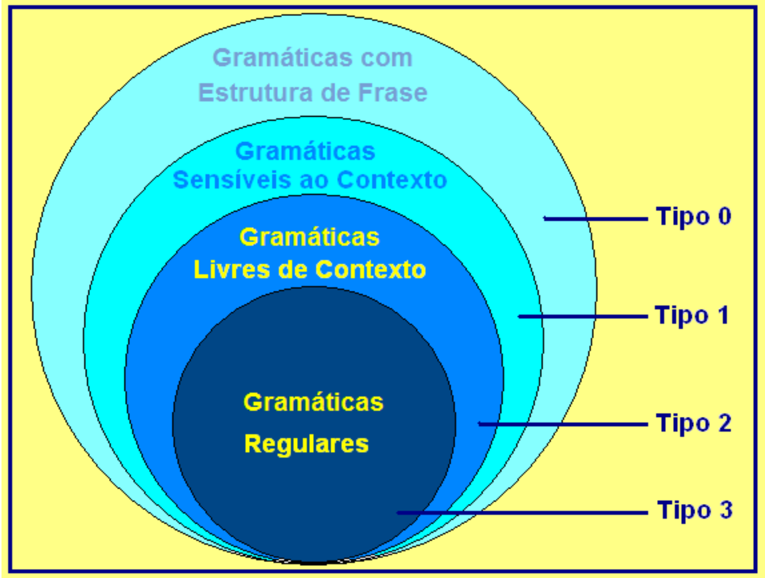
Hierarquia de Chmosky

- Classificação de gramáticas formais
- Descrita pelo linguísta americano *Noam Chmosky* em 1950
- Possui 4 níveis
- O nível 0: maior grau de liberdade
- O nível 3: menor grau de liberdade, mais restrito
- Um gramática do nível ***n*** também é uma gramática do ***n-1***



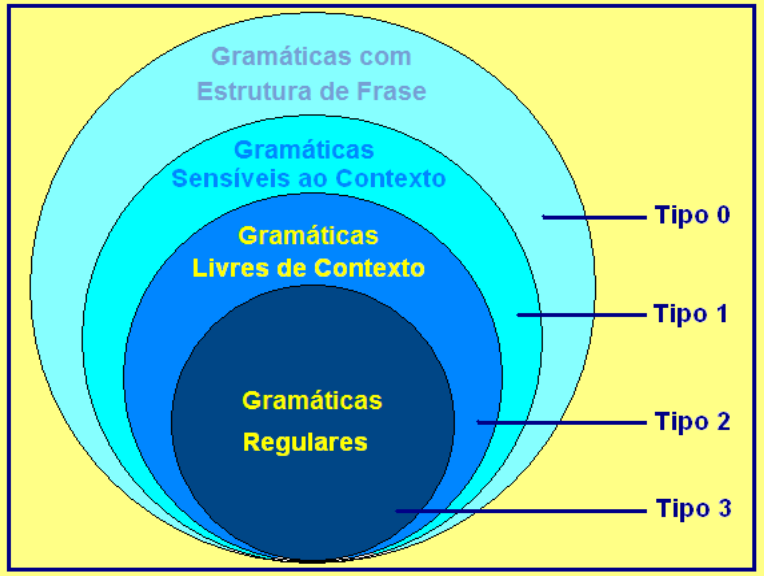
Hierarquia de Chmosky

- Tipo 0 : Gramáticas com estruturas de frase
- Tipo 1: Gram. Sensíveis ao contexto
- Tipo 2: Gram. Livres de contexto
- Tipo 3: Gram. Regulares



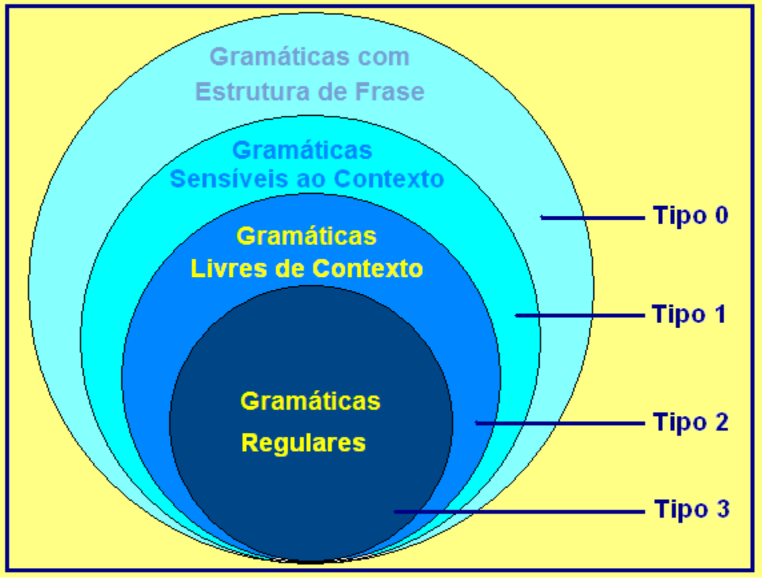
Hierarquia de Chmosky

- Tipo 0 : Gramáticas com estruturas de frase
  - Aquelas que não possuem limitações
- Tipo 1: Gram. Sensíveis ao contexto
- Tipo 2: Gram. Livres de contexto
- Tipo 3: Gram. Regulares



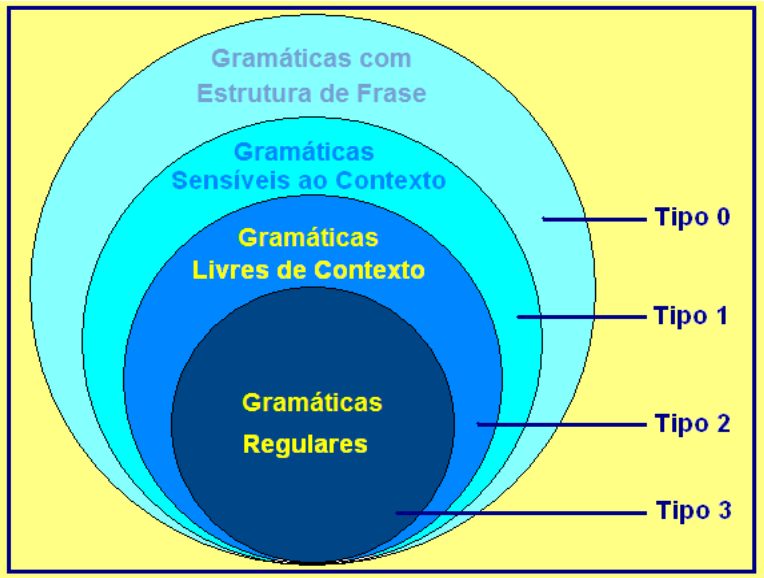
Hierarquia de Chmosky

- Tipo 0 : Gramáticas com estruturas de frase
- Tipo 1: Gram. Sensíveis ao contexto
  - Nenhuma das regras de produção pode reduzir o comprimento da forma sentencial que for substituida
  - Se $\alpha \rightarrow \beta$ então $|\alpha| <= |\beta|$
- Tipo 2: Gram. Livres de contexto
- Tipo 3: Gram. Regulares



Hierarquia de Chmosky

- Tipo 0 : Gramáticas com estruturas de frase
- Tipo 1: Gram. Sensíveis ao contexto
- Tipo 2: Gram. Livres de contexto
  - As regras tem apenas uma Variavel do lado esquerdo
  - Não pode ter terminal do lado esquerdo
  - São do tipo
    - $A \rightarrow \beta$
    - $Aa \rightarrow \beta$, não pode
- Tipo 3: Gram. Regulares



Prof José Rui

Hierarquia de Chmosky

- Tipo 0 : Gramáticas com estruturas de frase
- Tipo 1: Gram. Sensíveis ao contexto
- Tipo 2: Gram. Livres de contexto
- Tipo 3: Gram. Regulares
  - Além das restrições da tipo 2
  - Deve ser Linear à direita ou à esquerda
  - $A \rightarrow aB \mid a$
  - $B \rightarrow Ba \mid a$
  - $A \rightarrow ABa$, Não pode