Teoria da Computação

Linguagens Formais, Gramáticas e Autômatos

Mirtha Lina Fernández Venero
mirtha.lina@ufabc.edu.br
Sala 529-2, Bloco A

setembro 2017

Agenda

Introdução

Linguagens Formais

Operações sobre linguagens

Gramáticas

Autômatos

Bibliografia

Problema Computacional

Objeto matemático que pode ser definido como um conjunto de instâncias e a solução para cada uma delas

Exemplo: O problema de determinar se um número natural é primo pode ser representado como:

$$Primo = \{(1, yes), (2, yes), (3, yes), (4, no), (5, yes), (6, no), \ldots\}$$

Resolver um problema computacionalmente consiste em achar um algoritmo que para toda instância retorne uma solução.

Tipos de Problemas

- **de decisão**: A solução é *yes* ou *no*. **Exemplo**: *Primo*
- **busca**: Podem existir diferentes soluções para a mesma instância. **Exemplo**: Problema da fatoração: Dado um número n achar um fator primo não trivial

$$Fator = \{(4,2),(6,2),(6,3),(8,2),(9,3),(10,2),(10,5),\ldots\}$$

- **enumeração**: Achar todas as soluções duma instância

$$AllFators = \{(4,\{2\}),(6,\{2,3\}),(8,\{2\}),(9,\{3\}),(10,\{2,5\}),\ldots\}$$

- **de contagem**: Achar o número de soluções duma instância

$$\#Fators = \{(4,1),(6,2),(8,1),(9,1),(10,2),\ldots\}$$

- **otimização:** Achar a melhor/menor/maior das soluções

$$MinFator = \{(4,2),(6,2),(8,2),(9,3),(10,2),\ldots\}$$

Tipos de Problemas

A Teoria da Computação foca nos **problemas de decisão**:

- Os problemas de decisão podem ser representados usando somente o conjunto das instâncias para as quais a saída é *yes*, I_{yes}. Dessa forma, resolver o problema de decisão se reduce a resolver um problema de pertinença (membership problem), i.e. $i \in I_{yes}$
- Para qualquer outro tipo de problema P podemos achar um problema de decisão correspondente D tal que se não existe algoritmo para D, então não existe algoritmo para P
- É mais fácil trabalhar com conjuntos do que com funções
- Dependendo das características do conjunto, o problema de pertinença pode ser resolvido usando diferentes modelos de cômputo

Agenda

Introdução

Linguagens Formais

Operações sobre linguagens

Gramáticas

Autômatos

Bibliografia

Alfabeto, Palavra, Operações

- **Alfabeto**: conjunto *finito não vazio* de símbolos

  Exemplo: $\Sigma_1 = \{0, 1\}$, $\Sigma_2 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, $\Sigma_3 = \{a, b, c, \ldots, z\}$, $\Sigma_4 = \{ab, c1, zero\}$

Alfabeto, Palavra, Operações

- **Alfabeto**: conjunto *finito não vazio* de símbolos

  Exemplo: $\Sigma_1 = \{0, 1\}$, $\Sigma_2 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, $\Sigma_3 = \{a, b, c, \ldots, z\}$, $\Sigma_4 = \{ab, c1, zero\}$

- **Cadeia ou Palavra/Σ**: sequência *finita* de símbolos $\in \Sigma$

  Exemplo: 10101, 007, *ab*, *abzerozerozero*

Alfabeto, Palavra, Operações

- **Alfabeto**: conjunto *finito não vazio* de símbolos

  Exemplo: $\Sigma_1 = \{0, 1\}$, $\Sigma_2 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, $\Sigma_3 = \{a, b, c, \ldots, z\}$, $\Sigma_4 = \{ab, c1, zero\}$

- **Cadeia ou Palavra/Σ**: sequência *finita* de símbolos $\in \Sigma$

  Exemplo: 10101, 007, *ab*, *abzerozerozero*

- **Comprimento ou tamanho de palavra/Σ**: número de símbolos

  Exemplo: $|10101| = 5$, $|ab|_{\Sigma_3} = 2$, $|ab|_{\Sigma_4} = 1$, $|abzerozerozero|_{\Sigma_4} = 4$

Alfabeto, Palavra, Operações

- **Alfabeto**: conjunto *finito não vazio* de símbolos

  Exemplo: $\Sigma_1 = \{0, 1\}$, $\Sigma_2 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, $\Sigma_3 = \{a, b, c, \ldots, z\}$, $\Sigma_4 = \{ab, c1, zero\}$

- **Cadeia ou Palavra/Σ**: sequência *finita* de símbolos $\in \Sigma$

  Exemplo: 10101, 007, *ab*, *abzerozerozero*

- **Comprimento ou tamanho de palavra/Σ**: número de símbolos

  Exemplo: $|10101| = 5$, $|ab|_{\Sigma_3} = 2$, $|ab|_{\Sigma_4} = 1$, $|abzerozerozero|_{\Sigma_4} = 4$

- **Palavra vazia** ε: $|\varepsilon| = 0$

Alfabeto, Palavra, Operações

- **Alfabeto**: conjunto *finito não vazio* de símbolos

  Exemplo: $\Sigma_1 = \{0, 1\}$, $\Sigma_2 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, $\Sigma_3 = \{a, b, c, \ldots, z\}$, $\Sigma_4 = \{ab, c1, zero\}$

- **Operações sobre palavras/Σ:**
  - vw **concatenação** de v e w
  - v **sub-palavra** de w sse $xvy = w$
  - v **prefixo** de w sse $vy = w$
  - v **sufixo** de w sse $xv = w$
  - $\bar{v} = s_1 s_2 \ldots s_n$ **reverso** sse $v = s_n \ldots s_2 s_1$ (também v^r)

- **Propriedades da concatenação**
  - $|vw| = |v| + |w|$
  - $v\varepsilon = \varepsilon v = v$, $(xv)y = x(vy)$
  - $|\Sigma| > 1 \Rightarrow \exists v, \exists w,\ vw \neq wv$

Alfabeto, Palavra, Operações

- **Alfabeto**: conjunto *finito não vazio* de símbolos

  Exemplo: $\Sigma_1 = \{0, 1\}$, $\Sigma_2 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, $\Sigma_3 = \{a, b, c, \ldots, z\}$, $\Sigma_4 = \{ab, c1, zero\}$

- **Potência k de Σ:** Conjunto de palavras/Σ de tamanho k

  Exemplo: $\Sigma^0 = \{\varepsilon\}$ $\Sigma^1 = \Sigma = \{0, 1\}$, $\Sigma^2 = \{00, 01, 10, 11\}$

Alfabeto, Palavra, Operações

- **Alfabeto**: conjunto *finito não vazio* de símbolos

  Exemplo: $\Sigma_1 = \{0, 1\}$, $\Sigma_2 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, $\Sigma_3 = \{a, b, c, \ldots, z\}$, $\Sigma_4 = \{ab, c1, zero\}$

- **Potência k de Σ:** Conjunto de palavras/Σ de tamanho k

  Exemplo: $\Sigma^0 = \{\varepsilon\}$ $\Sigma^1 = \Sigma = \{0, 1\}$, $\Sigma^2 = \{00, 01, 10, 11\}$

- **Kleene closures/Σ**: conjunto de todas as palavras / Σ

$$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \ldots = \bigcup_{k=0}^{\infty} \Sigma^k$$

$$\Sigma^+ = \Sigma^* - \Sigma^0 = \bigcup_{k=1}^{\infty} \Sigma^k$$

Linguagem Formal

Qualquer $L \subseteq \Sigma^*$ é uma linguagem formal.

Exemplo: Sejam $\Sigma_1 = \{0, 1\}$, $\Sigma_2 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, $\Sigma_3 = \{a, b, c, \ldots, z\}$, exemplos de linguagens?

Linguagem Formal

Qualquer $L \subseteq \Sigma^*$ é uma linguagem formal.

- $\emptyset$, Σ, Σ^* são linguagens
- As linguagens podem ser finitas ou infinitas
- Σ^* é um conjunto contável, i.e. tem o mesmo tamanho que N
- $\mathcal{P}(\Sigma^*)$ é incontável (como R, a prova é por contradição usando a sequência característica de cada linguagem e o método da diagonalização de Cantor).

$$\begin{array}{rccccccccccc}
\Sigma^* = \{ & \varepsilon & , & 0 & , & 1 & , & 00 & , & 01 & , & 10 & , & 11 & , & 000 & , & 001 & , & \cdots & \} \\
A = \{ & & & 0 & , & & & 00 & , & 01 & , & & & & & 000 & , & 001 & , & \cdots & \} \\
\chi_A = & 0 & & 1 & & 0 & & 1 & & 1 & & 0 & & 0 & & 1 & & 1 & & \cdots &
\end{array}$$

Ver mais sobre a diagonalização no Sipser págs 183-188

Agenda

Introdução

Linguagens Formais

Operações sobre linguagens

Gramáticas

Autômatos

Bibliografia

Como especificar formalmente uma linguagem $L \subseteq \Sigma^*$?

- **Operações sobre conjuntos:** $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 - L_2$, L^c

Como especificar formalmente uma linguagem $L \subseteq \Sigma^*$?

- **Operações sobre conjuntos:** $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 - L_2$, L^c

  Exemplo: linguagem dos dígitos hexadecimais

Como especificar formalmente uma linguagem $L \subseteq \Sigma^*$?

- **Operações sobre conjuntos:** $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 - L_2$, L^c

  Exemplo: linguagem dos dígitos hexadecimais

- **Operações sobre linguagens/Σ:**

$$L_1 L_2, \quad \bar{L}, \quad L^0 = \{\varepsilon\}, \quad L^k = L^{k-1} L, \quad L^* = \bigcup_{k=0}^{\infty} L^k, \quad L^+ = L^* - L^0$$

  Exemplo: linguagem dos números hexadecimais, linguagem dos números pares, linguagem dos múltiplos de 5, linguagens dos identificadores Java

Como especificar formalmente uma linguagem $L \subseteq \Sigma^*$?

- **Operações sobre conjuntos:** $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 - L_2$, L^c

  Exemplo: linguagem dos dígitos hexadecimais

- **Operações sobre linguagens/Σ:**

$$L_1 L_2, \quad \bar{L}, \quad L^0 = \{\varepsilon\}, \quad L^k = L^{k-1} L, \quad L^* = \bigcup_{k=0}^{\infty} L^k, \quad L^+ = L^* - L^0$$

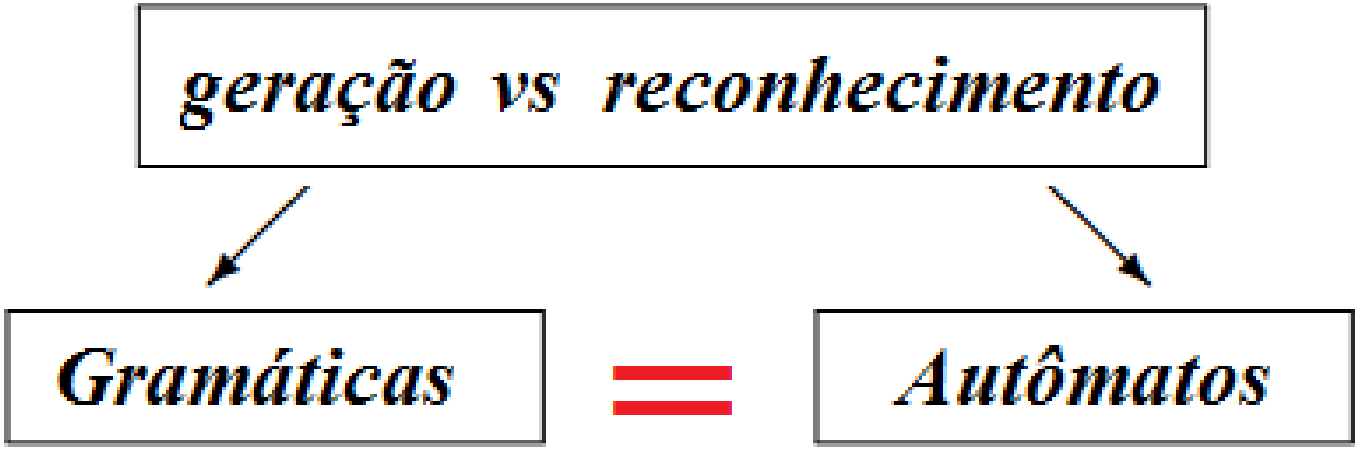
  Exemplo: linguagem dos números hexadecimais, linguagem dos números pares, linguagem dos múltiplos de 5, linguagens dos identificadores Java

  Exemplo: linguagem das cadeias que têm todas vogais? linguagem dos múltiplos de 3? linguagem das expressões aritméticas? linguagem dos números primos?

Especificação formal de linguagens

Surpreendentemente, pouquissimas linguagens podem ser especificadas usando operações de conjuntos.

Como especificar qualquer linguagem?



Universidade Federal do ABC

Agenda

Introdução

Linguagens Formais

Operações sobre linguagens

Gramáticas

Autômatos

Bibliografia

Gramáticas: $(\mathbf{N}, \mathbf{\Sigma}, \mathbf{S}, \mathbf{P})$

1. N (ou V) conjunto *finito* de símbolos **não-terminais***
2. Σ (ou T) conjunto *finito* de símbolos **terminais**
3. $S \in N$ **símbolo de começo**
4. $P \subseteq (N \cup \Sigma)^* N (N \cup \Sigma)^* \times (N \cup \Sigma)^*$ conjunto de **produções ou regras sintáticas** escritas $\alpha \to \beta$ $(\alpha \to \beta_1 | \beta_2 | \ldots | \beta_n)$

* Também chamados **variáveis** ou **categorias sintáticas**

Gramáticas: (N, Σ, S, P)

1. N (ou V) conjunto *finito* de símbolos **não-terminais***
2. Σ (ou T) conjunto *finito* de símbolos **terminais**
3. $S \in N$ **símbolo de começo**
4. $P \subseteq (N \cup \Sigma)^* N (N \cup \Sigma)^* \times (N \cup \Sigma)^*$ conjunto de **produções ou regras sintáticas** escritas $\alpha \to \beta$ $(\alpha \to \beta_1 | \beta_2 | \ldots | \beta_n)$

Exemplo:

$$G_1 = (\{A, S\}, \{0, 1, \ldots, 9\}, S, \{S \to AS | A, A \to 0 | 1 | \ldots | 9\})$$

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

$$G_3 = S \to aSBc, S \to abc, cB \to Bc, bB \to bb$$

* Também chamados **variáveis** ou **categorias sintáticas**

Gramáticas: $(\mathbf{N}, \mathbf{\Sigma}, \mathbf{S}, \mathbf{P})$

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

Gramáticas: (N, Σ, S, P)

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

  Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

Gramáticas: (N, Σ, S, P)

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

  Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

  Exemplo: $S \to_G 0A1$

Gramáticas: (N, Σ, S, P)

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

  Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

  Exemplo: $S \to_G 0A1$

Gramáticas: (N, Σ, S, P)

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

  Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

  Exemplo: $S \to_G 0A1 \to_G 00A11$

Universidade Federal do ABC

Gramáticas: (N, Σ, S, P)

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

  Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

  Exemplo: $S \to_G 0A1 \to_G 00A11$

Gramáticas: (N, Σ, S, P)

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

  Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

  Exemplo: S $\to_G$ 0A1 $\to_G$ 00A11

Gramáticas: (N, Σ, S, P)

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

  Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

  Exemplo: S $\to_G$ 0A1 $\to_G$ 00A11 $\to_G$ 000A111

Gramáticas: (N, Σ, S, P)

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

  Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

  Exemplo: $S \to_G 0A1 \to_G 00A11 \to_G 000A111$

Universidade Federal do ABC

Gramáticas: (N, Σ, S, P)

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

  Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

  Exemplo: $S \to_G 0A1 \to_G 00A11 \to_G 000A111$

Universidade Federal do ABC

Gramáticas: $(\mathbf{N}, \mathbf{\Sigma}, \mathbf{S}, \mathbf{P})$

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

  Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

  Exemplo: S $\to_G$ 0A1 $\to_G$ 00A11 $\to_G$ 000A111 $\to_G$ 000ε111

Universidade Federal do ABC

Gramáticas: (N, Σ, S, P)

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

  Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

  Exemplo: $S \to_G 0A1 \to_G 00A11 \to_G 000A111 \to_G 000111$

Gramáticas: (N, Σ, S, P)

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

Exemplo: $S \to_G 0A1 \to_G 00A11 \to_G 000A111 \to_G 000111$

Derivação: $\gamma \to_G^* \eta$ sse $\gamma = \eta$ ou $\gamma \to_G \theta \to_G^* \eta$

Gramáticas: (N, Σ, S, P)

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

Exemplo: $S \to_G 0A1 \to_G 00A11 \to_G 000A111 \to_G 000111$

Derivação: $\gamma \to_G^* \eta$ sse $\gamma = \eta$ ou $\gamma \to_G \theta \to_G^* \eta$

Exemplo: $S \to_G^* 000111$

Gramáticas: (N, Σ, S, P)

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

  Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

  Derivação: $\gamma \to_G^* \eta$ sse $\gamma = \eta$ ou $\gamma \to_G \theta \to_G^* \eta$

Linguagem gerada por G: $\mathbf{L}(G) = \{w \in \Sigma^* \mid S \to_G^* w\}$

Gramáticas: $(\mathbf{N}, \mathbf{\Sigma}, \mathbf{S}, \mathbf{P})$

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

Derivação: $\gamma \to_G^* \eta$ sse $\gamma = \eta$ ou $\gamma \to_G \theta \to_G^* \eta$

Linguagem gerada por G: $\mathbf{L}(G) = \{w \in \Sigma^* \mid S \to_G^* w\}$

Exemplo: $\mathbf{L}(G_2) = \{0^n 1^n \mid n > 0\}$

Gramáticas: $(\mathbf{N}, \mathbf{\Sigma}, \mathbf{S}, \mathbf{P})$

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

Processo de Geração:

- Começar pelo **símbolo de começo**
- Aplicar produções até chegar a (ou derivar) uma palavra de Σ

Passo de derivação: $\gamma\ \alpha\ \eta \to_G \gamma\ \beta\ \eta$ sse $\alpha \to \beta \in P$

Derivação: $\gamma \to_G^* \eta$ sse $\gamma = \eta$ ou $\gamma \to_G \theta \to_G^* \eta$

Linguagem gerada por G: $\mathbf{L}(G) = \{w \in \Sigma^* \mid S \to_G^* w\}$

Equivalência de Gramáticas: $G \cong G'$ sse $\mathbf{L}(G) = \mathbf{L}(G')$

Classificação de Gramáticas

- **Irrestrita, universal ou de estrutura de frase**
- **Sensível ao contexto**: toda regra é da forma $\alpha \rightarrow \beta$ com $|\alpha| \leq |\beta|$ [1] exceto $S \rightarrow \varepsilon$ caso $\varepsilon \in L(G)$ [2]
- **Livre de contexto**: toda regra é da forma A $\rightarrow \beta$
- **Lineal à direita**: toda regra é da forma A $\rightarrow w\ B$[3]

[1] De forma equivalente: toda regra é da forma $\gamma\ A\ \eta \rightarrow \gamma\ \beta\ \eta$ com $\beta \neq \varepsilon$.
[2] Neste caso S não pode aparecer na parte direita de nenhuma produção.
[3] Neste caso $w \in \Sigma^*$ e $B \in N \cup \{\varepsilon\}$

Classificação de Gramáticas

- **Irrestrita, universal ou de estrutura de frase**
- **Sensível ao contexto**: toda regra é da forma $\alpha \rightarrow \beta$ com $|\alpha| \leq |\beta|$ [1] exceto $S \rightarrow \varepsilon$ caso $\varepsilon \in L(G)$ [2]
- **Livre de contexto**: toda regra é da forma A $\rightarrow$ β
- **Lineal à direita**: toda regra é da forma A $\rightarrow w\ B$[3]

Exemplo:

$$G_1 = (\{A, S\}, \{0, 1, \ldots, 9\}, S, \{S \rightarrow AS|A, A \rightarrow 0|1|\ldots|9\})$$

$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \rightarrow 0A1, 0A \rightarrow 00A1, A \rightarrow \varepsilon\})$$

[1] De forma equivalente: toda regra é da forma $\gamma\ A\ \eta \rightarrow \gamma\ \beta\ \eta$ com $\beta \neq \varepsilon$.
[2] Neste caso S não pode aparecer na parte direita de nenhuma produção.
[3] Neste caso $w \in \Sigma^*$ e $B \in N \cup \{\varepsilon\}$

Classificação de Linguagens

- **Tipo 0**: existe uma gramática G tq $\mathbf{L}(G) = L$
- **Tipo 1 ou Sensível ao Contexto**: se $\mathbf{L}(G) = L$ para alguma gramática G sensível ao contexto
- **Tipo 2 ou Livre de contexto**: se $\mathbf{L}(G) = L$ para alguma gramática G livre de contexto
- **Tipo 3 ou Regular**: se $\mathbf{L}(G) = L$ para alguma gramática G lineal à direita (ou esquerda)

Classificação de Linguagens

- **Tipo 0**: existe uma gramática G tq $\mathbf{L}(G) = L$
- **Tipo 1 ou Sensível ao Contexto**: se $\mathbf{L}(G) = L$ para alguma gramática G sensível ao contexto
- **Tipo 2 ou Livre de contexto**: se $\mathbf{L}(G) = L$ para alguma gramática G livre de contexto
- **Tipo 3 ou Regular**: se $\mathbf{L}(G) = L$ para alguma gramática G lineal à direita (ou esquerda)

Classificar uma linguagem não é simples

1. escrever G e provar que $\mathbf{L}(G) = L \Leftrightarrow \mathbf{L}(G) \subseteq L$ e $\mathbf{L}(G) \supseteq L$
2. obter a classificação maior: a linguagem é tipo i **mas não é** de tipo $i + 1$

Classificação de Linguagens

- **Tipo 0**: existe uma gramática G tq $\mathbf{L}(G) = L$
- **Tipo 1 ou Sensível ao Contexto**: se $\mathbf{L}(G) = L$ para alguma gramática G sensível ao contexto
- **Tipo 2 ou Livre de contexto**: se $\mathbf{L}(G) = L$ para alguma gramática G livre de contexto
- **Tipo 3 ou Regular**: se $\mathbf{L}(G) = L$ para alguma gramática G lineal à direita (ou esquerda)

Exemplo: Como classificar $L(G_1), L(G_2), L(G_3)$?

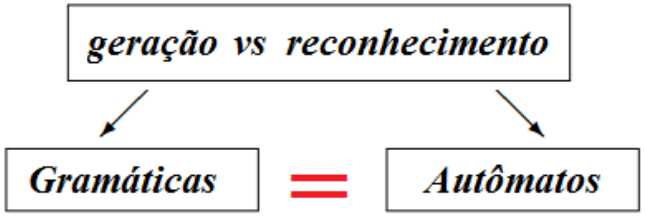
$$G_1 = (\{A, S\}, \{0, 1, \ldots, 9\}, S, \{S \to AS|A, A \to 0|1|\ldots|9\})$$

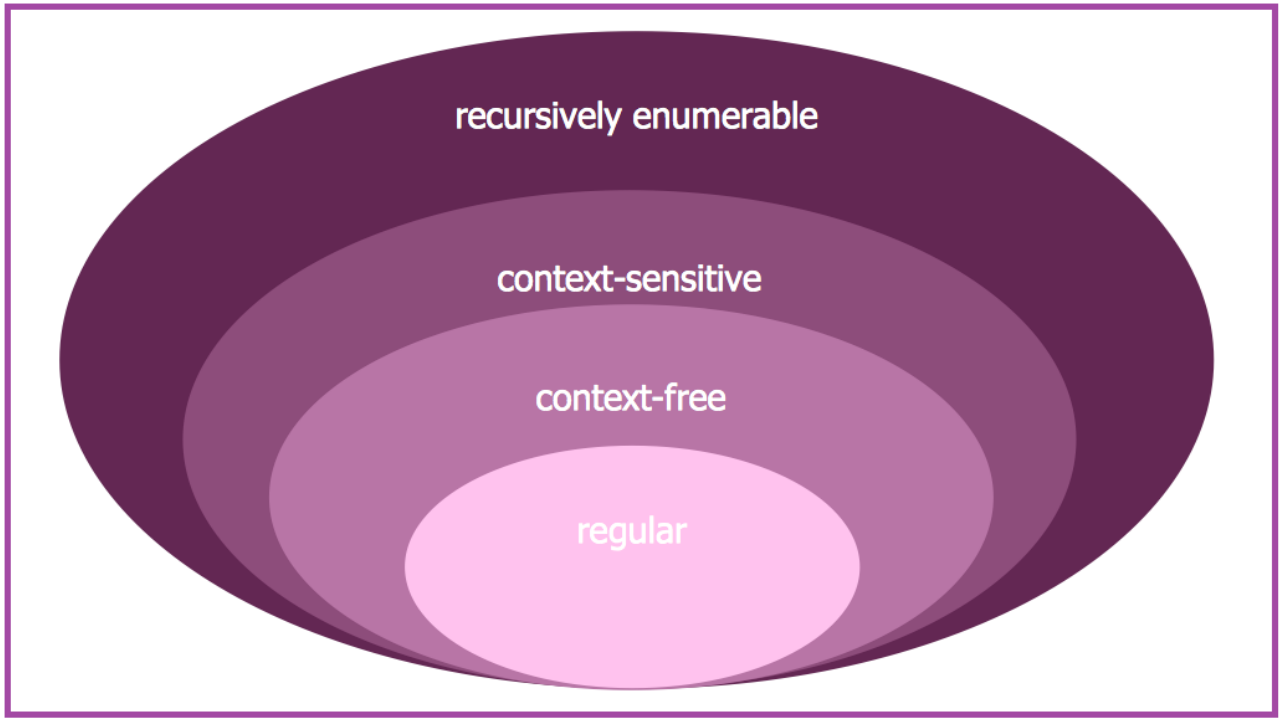
$$G_2 = (\{A, S\}, \{0, 1\}, S, \{S \to 0A1, 0A \to 00A1, A \to \varepsilon\})$$

$$G_3 = S \to aSBc, S \to abc, cB \to Bc, bB \to bb$$

Universidade Federal do ABC

Classificação de Linguagens
Hierarquia de Chomsky





http://www.conceptdraw.com/How-To-Guide/venn-diagram-problem-solving-examples-computer

Agenda

Introdução

Linguagens Formais

Operações sobre linguagens

Gramáticas

Autômatos

Bibliografia

Especificação formal de linguagens

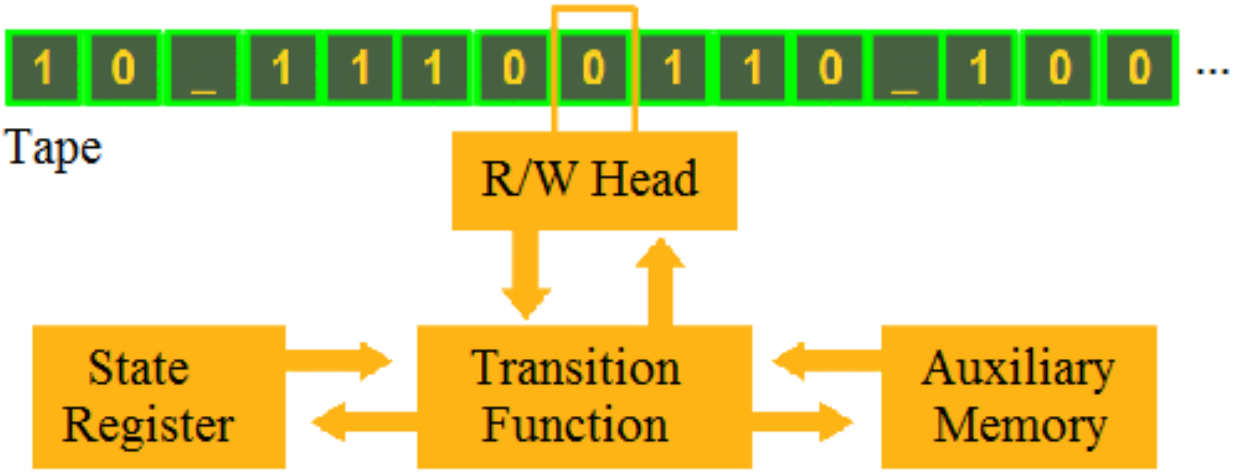
Autômatos: $(Q, \Sigma, \Gamma, q_0, \delta, F)$

- Q conjunto *finito não vazio* de estados
- Σ alfabeto de entrada
- Γ conjunto de símbolos
- $q_0 \in Q$ estado inicial
- δ uma função de transição de estados
- $F \subseteq Q$ conjunto de estados finais
- outras componentes dependendo do tipo de autômato

Universidade Federal do ABC

Especificação formal de linguagens

Autômatos: $(Q, \Sigma, \Gamma, q_0, \delta, F)$

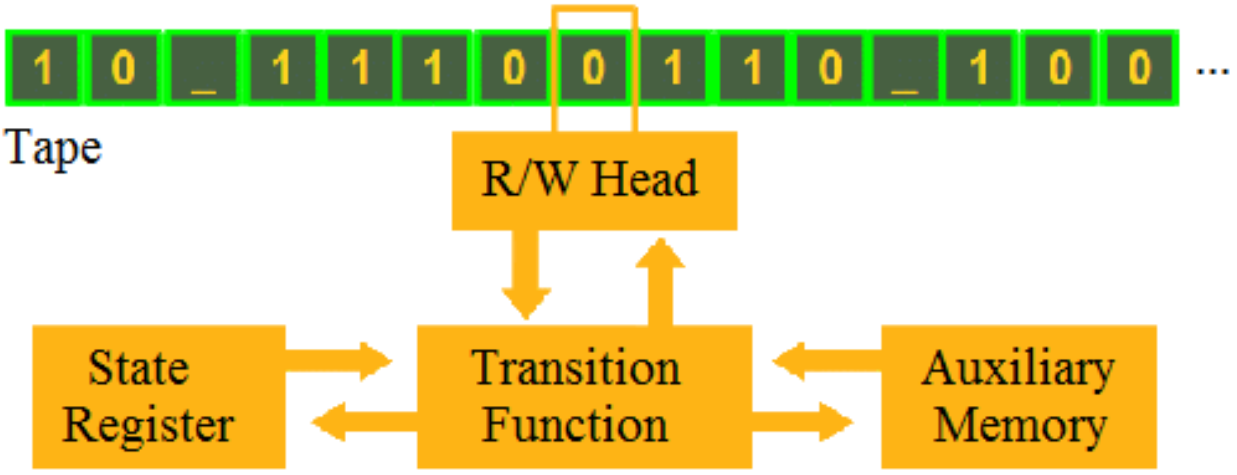


- finito
- de pilha
- lineal limitado
- Máquina de Turing

Universidade Federal do ABC

Especificação formal de linguagens

Autômatos: $(Q, \Sigma, \Gamma, q_0, \delta, F)$

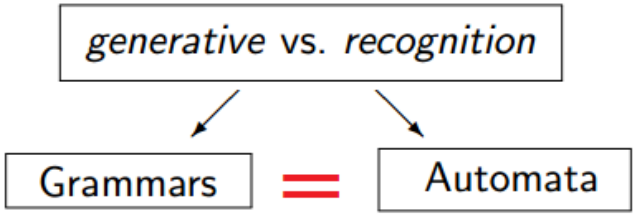


- **finito**
- **de pilha**
- **lineal limitado**
- **Máquina de Turing**

Processo de Reconhecimento:

- Começar pelo **estado inicial**
- Ler cada símbolo da cadeia e mudar de estado dependendo δ até chegar num estado aceitação ou rejeição

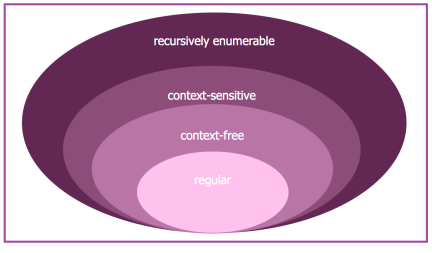
Classificação de Linguagens
Hierarquia de Chomsky



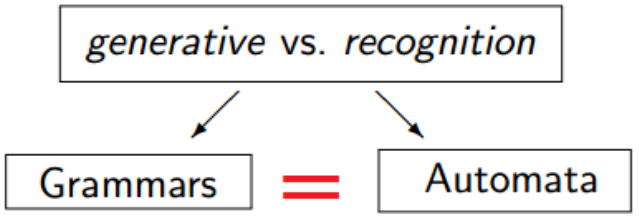
Tipos	Gramáticas	Autômatos	Linguagens
0	não restritiva	Máquina de Turing	**RE**
1	sensível ao contexto	lineal limitado	**SC**
2	livre de contexto	de pilha não determinístico	**LC**
3	lineal à direita	finito	**REG**

$$\textbf{FIN} \subset \textbf{REG} \subset \textbf{LC} \subset \textbf{SC} \subset \textbf{REC} \subset \textbf{RE} \subset \wp(\Sigma^*)$$

FIN = linguagens finitas, **REG** = linguagens regulares,
LC = linguagens livres de contexto,
SC = linguagens sensíveis ou dependentes do contexto,
REC = linguagens recursivas ou decidíveis,
RE = linguagens recursivamente enumeráveis ou semi-decidíveis,
$\wp(\Sigma^*)$ = todas as linguagens sobre um alfabeto Σ
$\wp(\Sigma^*)$ - **RE** = linguagens não decidíveis



Especificação formal de linguagens



Resumo

- Gramáticas e autômatos proporcionam métodos formais, finitos e compactos para especificar linguagens (mesmo que infinitas)
- Ambos têm o mesmo poder de especificação
- Gramáticas mais adequadas para humanos; autômatos mais adequados para máquinas
- Ambos indispensáveis para construir compiladores

Universidade Federal do ABC

Agenda

Introdução

Linguagens Formais

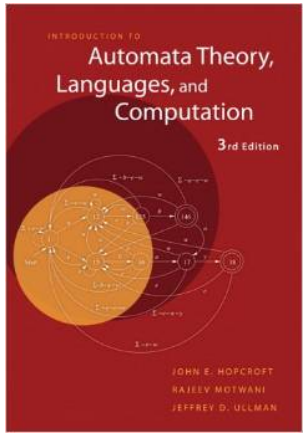
Operações sobre linguagens

Gramáticas

Autômatos

Bibliografia

Bibliografia



1. **Introduction to Automata Theory, Languages, and Computation** (3rd Edition). J. Hopcroft, R. Motwani and J. Ullman. Addison-Wesley, 2006
2. **Compilers: Principles, Techniques, and Tools** (2nd Edition). Alfred Aho, Monica Lam, Ravi Sethi, and Jeffrey Ullman. Addison-Wesley, 2006
3. Introduction to the Theory of Computation. M. Sipser
4. Theory of Computation: Formal Languages, Automata, and Complexity by J. Glenn Brookshear
5. Formal Language: A Practical Introduction, A. B. Webber
6. Linguagens Formais e Autômatos, P. B. Menezes