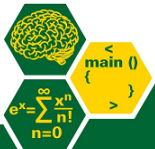




Universidade Federal do ABC

CMCC

Centro de Matemática, Computação e Cognição



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



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

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

- ▶ **enumeração:** Achar todas as soluções de cada instância

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

- ▶ **de contagem:** Achar o número de soluções de cada instância

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

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

$$MenorFator = \{(4, 2), (6, 2), (8, 2), (9, 3), (10, 2), \dots\}$$



Importância dos 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 pertinência (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 pertinência pode ser resolvido usando diferentes modelos de cômputo; alguns bem simples, outros mais complexos



Importância dos 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 pertinência (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 pertinência pode ser resolvido usando diferentes modelos de cômputo; alguns bem simples, outros mais complexos

Como representar um conjunto de instâncias?



Agenda

Introdução

Linguagens Formais

Operações sobre linguagens

Gramáticas

Autômatos

Bibliografia



Alfabeto, Palavra, Operações

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

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



Alfabeto, Palavra, Operações

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

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

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

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

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

- ▶ **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 $xv = w$
- ▶ $\bar{v} = s_1s_2 \dots s_n$ **reverso** sse $v = s_n \dots s_2s_1$ (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: $\Sigma_1 = \{0, 1\}$, $\Sigma_2 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$,
 $\Sigma_3 = \{a, b, c, \dots, z\}$, $\Sigma_4 = \{ab, c1, zero\}$

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

- ▶ **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

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

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



Linguagem Formal

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

Exemplo: Sejam $\Sigma_1 = \{0, 1\}$, $\Sigma_2 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$,
 $\Sigma_3 = \{a, b, c, \dots, z\}$, 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 \mathbb{N}
- ▶ $\mathcal{P}(\Sigma^*)$ é incontável (como \mathbb{R} , a prova é por contradição usando a sequência característica de cada linguagem e o método da diagonalização de Cantor).

$$\begin{aligned}\Sigma^* &= \{ \epsilon, 0, 1, 00, 01, 10, 11, 000, 001, \dots \} \\ A &= \{ \quad 0, \quad 00, 01, \quad \quad 000, 001, \dots \} \\ \chi_A &= \quad 0 \quad 1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad \dots\end{aligned}$$

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 :**

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

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 :**

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

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

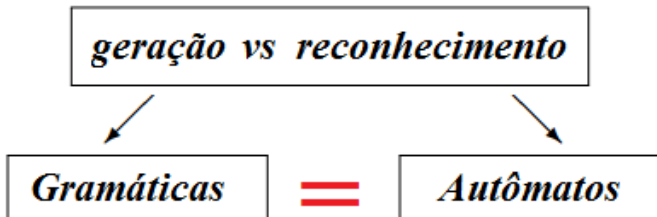
Exemplo: linguagem das cadeias que têm todas vogais?
linguagem dos múltiplos de 3? linguagem das expressões aritméticas? linguagem dos números primos?



Especificação formal de linguagens

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

Como especificar qualquer linguagem?





Agenda

Introdução

Linguagens Formais

Operações sobre linguagens

Gramáticas

Autômatos

Bibliografia



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 $\alpha \rightarrow \beta$ ($\alpha \rightarrow \beta_1 | \beta_2 | \dots | \beta_n$)

* 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 $\alpha \rightarrow \beta$ ($\alpha \rightarrow \beta_1 | \beta_2 | \dots | \beta_n$)

Exemplo:

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

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

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

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



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

Exemplo: $S \rightarrow_G 0A1$



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

Exemplo: $S \rightarrow_G 0A1$



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

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



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

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



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

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



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

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



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

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



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

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



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

Exemplo: $S \rightarrow_G 0A1 \rightarrow_G 00A11 \rightarrow_G 000A111 \rightarrow_G 000\varepsilon111$



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

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



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

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

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



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

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

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

Exemplo: $S \rightarrow_G^* 000111$



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

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

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

Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

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

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

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



Gramáticas: (N, Σ, S, P)

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

Processo de Geração:

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

Passo de derivação: $\gamma \alpha \eta \rightarrow_G \gamma \beta \eta$ sse $\alpha \rightarrow \beta \in P$

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

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

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



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 $A \rightarrow \beta$
- ▶ **Lineal à direita:** toda regra é da forma $A \rightarrow w B^3$

Exemplo:

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

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

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 $L(G) = L$
- ▶ **Tipo 1 ou Sensível ao Contexto:** se $L(G) = L$ para alguma gramática G sensível ao contexto
- ▶ **Tipo 2 ou Livre de contexto:** se $L(G) = L$ para alguma gramática G livre de contexto
- ▶ **Tipo 3 ou Regular:** se $L(G) = L$ para alguma gramática G lineal à direita (ou esquerda)



Classificação de Linguagens

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

Classificar uma linguagem não é simples

1. escrever G e provar que $L(G) = L \Leftrightarrow L(G) \subseteq L$ e $L(G) \supseteq L$
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 $L(G) = L$
- ▶ **Tipo 1 ou Sensível ao Contexto:** se $L(G) = L$ para alguma gramática G sensível ao contexto
- ▶ **Tipo 2 ou Livre de contexto:** se $L(G) = L$ para alguma gramática G livre de contexto
- ▶ **Tipo 3 ou Regular:** se $L(G) = L$ para alguma gramática G lineal à direita (ou esquerda)

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

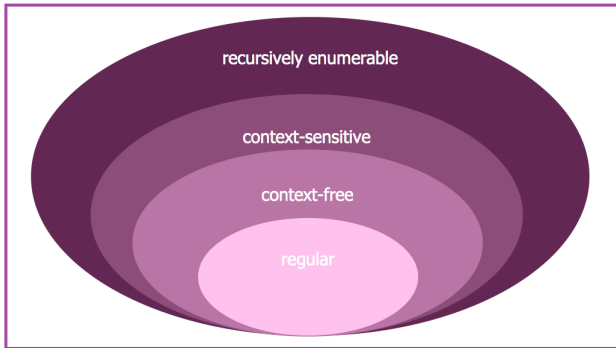
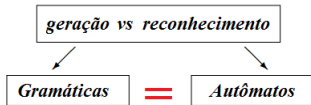
$$G_1 = (\{A, S\}, \{0, 1, \dots, 9\}, S, \{S \rightarrow AS | A, A \rightarrow 0|1|\dots|9\})$$

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

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

Classificação de Linguagens

Hierarquia de Chomsky





Agenda

Introdução

Linguagens Formais

Operações sobre linguagens

Gramáticas

Autômatos

Bibliografia



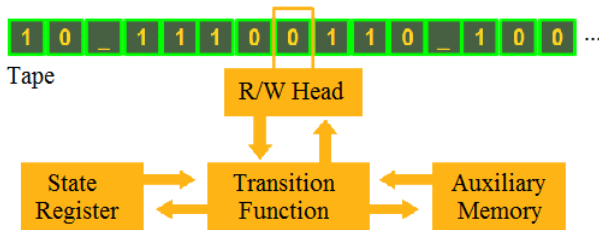
Especificação formal de linguagens

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

Especificação formal de linguagens

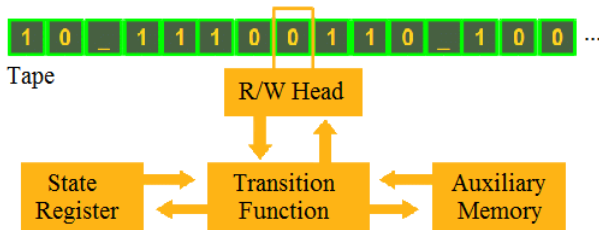
Autômatos: $(Q, \Sigma, \Gamma, q_0, \delta, F)$



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

Especificação formal de linguagens

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



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

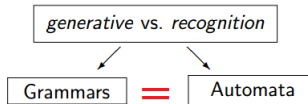
Processo de Reconhecimento:

- ▶ 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

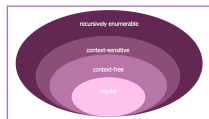
Hierarquia de Chomsky



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

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

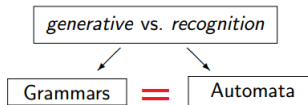
FIN = linguagens finitas, **REG** = linguagens regulares,
LC = linguagens livres de contexto,
SC = linguagens sensíveis ou dependentes do contexto,
REC = linguagens recursivas ou decidíveis,
RE = linguagens recursivamente enumeráveis ou semi-decidíveis,
 $\wp(\Sigma^*)$ = todas as linguagens sobre um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



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

