Linguagens Formais e Autômatos

Linguagens Livres de Contexto

Eduardo Furlan Miranda

Baseado em: GARCIA, A. de V.; HAEUSLER, E. H. Linguagens Formais e Autômatos. Londrina: EDA, 2017.

Hierarquia de Chomsky

R $a^n b$ ε, b, ab, aab
LC $a^n b^n$ ab, aabb
SC $a^n b^n c^n$ abc, aabbcc
I a^{2^n} a, aa, aaaa

apenas estas

Gramáticas	Regras	Ex. de linguagens geradas
GR (tipo 3) Regulares	A→aB, A→b, (A→ε, se permitido, apenas para o símbolo inicial) A, B ∈ V (variáveis) a, b ∈ T (terminais)	{ ε, b, ab, aab, aaab, ... } = { $a^n b$ | $n \geq 0$} ∪ {ε}
GLC (tipo 2) Livres de Contexto	$A \rightarrow \alpha$ $A \in V$, $\alpha \in (V \cup T)^*$ A: 1 única variável	{ ab, aabb, aaabbb, aaaabbbb, ... } = { $a^n b^n$ | $n > 0$ }
GSC (tipo 1) Sensíveis ao Contexto	$\alpha \rightarrow \beta$ $\alpha \in (V \cup T)^+$, $\beta \in (V \cup T)^*$ $\|\alpha\| \leq \|\beta\|$ S → ε, se S não aparece do lado direito de nenhuma regra	{ abc, aabbcc, aaabbbccc, ... } = { $a^n b^n c^n$ | $n > 0$ }
GI (tipo 0) Irrestrita ou geral	$\alpha \rightarrow \beta$ $\alpha, \beta \in (V \cup T)^*$ α: pelo menos 1 símbolo de V	{ a, aa, aaaa, aaaaaaaa, ... } = { a^{2^n} | $n \geq 0$}

Linguagens Livres de Contexto (LLC)

- Se uma Linguagem L é gerada por uma Gramática Livre de Contexto (GLC) G ,
  - dizemos que L é uma Linguagem Livre de Contexto (LLC)

- **Ex. 1**:
  - linguagem sobre o alfabeto $\Sigma = \{ a , b \}$, definida como $L_R = \{ w \mid w^R = w \}$, onde
    - w^R é a cadeia w revertida (de trás para frente)
    - L_R é linguagem cuja cadeia revertida é igual à cadeia original

      "L de R"
  - L_R é uma LLC porque é gerada pela GLC: $S \rightarrow aSa \mid bSb \mid a \mid b \mid \varepsilon$
    - $S \rightarrow aSa \rightarrow abSb \rightarrow abbSb \rightarrow abbbba$:
      - Linguagem gerada: "abbbba"

LLC

- Visto anteriormente: se L é uma Linguagem Regular (LR), então seu complemento, $\overline{L}$, também o é

- Ex.: $L = \{ w \in \{ a , b \}^* \mid w$ contém um número par de $a's \}$
- O complemento de L é definido como todas as cadeias que não estão em L.
Assim, $L^c = \{ w \in \{ a , b \}^* \mid w$ contém um número ímpar de $a's \}$.

- Dada a GLC que gera a linguagem $\overline{L}_R$
  - S → aSa | bSb | aAb | bAa
  - A → aAa | bAb | aAb | bAa | a | b | ε

cadeia com terminais e variáveis obtida aplicando as regras de produção da gramática

- A forma sentencial gerada possui exatamente uma variável: A ou S
  - **Ex.:** a partir do símbolo inicial S , podemos derivar:
    S → aSa → aaAba

é uma forma sentencial, pois contém terminais (a, a, b, a) e uma variável (A)

(continua)

- Uma derivação a partir de S apresentará até determinada regra apenas S como variável na forma sentencial ($S \rightarrow aSa$),
  - mas depois da aplicação de uma das regras $S \rightarrow aAb$ ou $S \rightarrow bAa$, as formas sentenciais terão apenas A como variável ($S \rightarrow aSa \rightarrow aaAba$)

- O funcionamento da gramática é tal que as formas sentenciais que têm a variável S são simétricas, ou seja
  - são iguais quando lidas de trás para frente,
    - enquanto que as formas sentenciais que têm a variável A são assimétricas, ou seja
      - são diferentes quando lidas de trás para a frente

Exemplo

- Dado o alfabeto $\Sigma = \{ a, b \}$, considere a linguagem sobre Σ definida como $L_1 = \{a^n b^{2n} \mid n \geq 1 \}$

- Podemos dizer que L_1 é uma LLC porque é gerada pela GLC:
  - $S \rightarrow aSbb$
  - $S \rightarrow abb$

- Ex.: para gerar aabbbb, podemos usar a derivação
  - $S \Rightarrow aSbb \Rightarrow aabbbb$

LLC são fechadas em relação à união

- Se pegarmos duas linguagens livres de contexto (LLCs), a união dessas duas linguagens resultará em outra linguagem livre de contexto (são fechadas em relação à união)
  - Em outras palavras, a propriedade de ser livre de contexto é preservada quando duas dessas linguagens são unidas

- Isso significa que, dadas duas LLC, L_1 e L_2, a linguagem $L_1 \cup L_2$ também será livre de contexto

- Se L_1 é gerada por uma GLC G_1 com símbolo inicial S_1 , e L_2 por uma GLC G_2 com símbolo inicial S_2 ,
  - podemos criar um novo símbolo inicial S e adicionar as regras $S \rightarrow S_1 \mid S_2$ às regras das duas gramáticas, obtendo assim uma GLC G_3 que gera $L_1 \cup L_2$

(continua)

Exemplo - união

- $L_1 : \{a^n b^n \mid n \geq 0\}$; $G_1 : S_1 \rightarrow aS_1b \mid \varepsilon$
- $L_2 : \{c^n d^n \mid n \geq 0\}$; $G_2 : S_2 \rightarrow cS_2d \mid \varepsilon$
- Para unir L_1 e L_2, criamos um novo símbolo inicial S e adicionamos as regras $S \rightarrow S_1 \mid S_2$ às regras de G_1 e G_2 :
  $L_1 \cup L_2 : \{a^n b^n \mid n \geq 0\} \cup \{c^n d^n \mid n \geq 0\}$
- G_3 :
  - $S \rightarrow S_1 \mid S_2$
  - $S_1 \rightarrow aS_1b \mid \varepsilon$
  - $S_2 \rightarrow cS_2d \mid \varepsilon$
- Dessa forma, G_3 gera a união de L_1 e L_2

LLC e concatenação

- Dado duas LLC, L_1 e L_2

- A sua concatenação é uma LLC
  - $L_1 \circ L_2 = \{ \omega_1 \circ \omega_2 \mid \omega_1 \in L_1 \text{ e } \omega_2 \in L_2 \}$

- Se L_1 é gerada por uma GLC G_1 com símbolo inicial S_1 , e L_2 é gerada por uma GLC G_2 com símbolo inicial S_2 ,
  - podemos criar um novo símbolo inicial S , acrescentar as regras $S \to S_1S_2$ à união das regras das duas gramáticas e obter assim uma GLC G_3 que gera $L_1 \circ L_2$

Exemplo - concatenação

- $L_1 = \{ a^n b^{2n} \mid n \geq 1 \}$ é gerada pela GLC :
  - $S_1 \rightarrow aS_1bb \mid abb$
- $L_2 = \{ a^{2n} b^n \mid n \geq 1 \}$ é gerada pela GLC :
  - $S_2 \rightarrow aaS_2b \mid aab$
- Portanto, a linguagem $L_1 \circ L_2$ pode ser gerada pela gramática
  - $S \rightarrow S_1S_2$
  - $S_1 \rightarrow aS_1bb \mid abb$
  - $S_2 \rightarrow aaS_2b \mid aab$

LLC e fecho de Kleene

- Se L_1 é uma LLC , então L_1^* também o é

- Este fato pode ser verificado através de uma construção com gramáticas

- Se L_1^* é gerada por uma GLC G_1 com símbolo inicial S_1 ,
  - para criarmos uma gramática que gera L_1^* basta criarmos um novo símbolo inicial S , e acrescentarmos as regras $S \rightarrow SS_1 \mid \varepsilon$ à gramática original

Exemplo - fecho de Kleene

- Seja $L_1 = \{ a^n b^{2n} \mid n \geq 1 \}$

- A linguagem é gerada pela GLC:
  - $S_1 \rightarrow aS_1bb \mid abb$

- Portanto, a linguagem L_1^* pode ser gerada pela gramática:
  - $S \rightarrow SS_1 \mid \varepsilon$
  - $S_1 \rightarrow aS_1bb \mid abb$

Gramáticas Livres de Contexto Sem Regras Nulas (GSRN)

- Definida por Hopcroft; Ullman (1969) como uma GLC onde
  - a cadeia vazia ε não é permitida no lado direito da regra,
    - exceto no caso da regra $S \rightarrow \varepsilon$, onde S é o símbolo inicial, e S não ocorre no lado direito de outra regra

- **Ex.:** seja a linguagem $L = \{a^n\, b^n \mid n \geq 0\}$, onde a quantidade de 'a's é igual à quantidade de 'b's. A GLC correspondente pode ser definida por $G : S \rightarrow aSb \mid \varepsilon$
  - Para transformá-la em GSRN: $S \rightarrow aSb \mid ab$
    - removemos a regra nula $S \rightarrow \varepsilon$
      - e adaptamos a gramática para preservar a igualdade entre a quantidade de 'a's e 'b's

GSRN - definição

- Dizemos que uma GLC $G = (V, T, P, S)$ é uma Gramática Livre de Contexto Sem Regras Nulas (GSRN) se todas as suas regras são da forma:
  - $A \rightarrow \alpha$, onde: $A \in V$ e $\alpha \in (V \cup T)^+$ ← fecho positivo
  - e desde que S (símbolo inicial) não esteja do lado direito de uma regra, é permitido $S \rightarrow \varepsilon$

$\alpha \in (V \cup T)^+$ define que α é uma string composta por um ou mais símbolos, sendo cada símbolo um terminal ou variável da gramática, excluindo a possibilidade de α ser a string vazia

GSRN

- A definição de Hopcroft; Ullman (1969) não altera o poder de expressividade das GLC, isto é,
  - se uma linguagem L é gerada por uma GLC G ,
  - então a mesma linguagem é gerada por uma GSRN G^1
- Existe um algoritmo para dada uma GLC G obtermos uma GSRN G^1 equivalente
  - O primeiro passo deste algoritmo é identificar as variáveis que podem gerar a cadeia vazia, chamadas de símbolos nulificáveis

Exemplo - nulificáveis

- A gramática:
  - $S \rightarrow aA$
  - $A \rightarrow aB \mid B$
  - $B \rightarrow b \mid \varepsilon$
- Possui as
  - variáveis {S, A, B}   e as
  - variáveis nulificáveis {A, B}
    - Derivando B, chegamos em ε : $B \Rightarrow^* \varepsilon$
    - O mesmo ocorre para A: $A \Rightarrow^* \varepsilon$

- A pode derivar *B* (pela regra $A \rightarrow B$)
- *B* já sabemos que é nulificável
- *A* também pode derivar a cadeia vazia via $A \rightarrow B \Rightarrow \varepsilon$
- Portanto, *A* é nulificável porque pode derivar *B*, e *B* pode derivar ε

GSRN equivalente

- Uma vez que sabemos quais são os símbolos nulificáveis de uma GLC, podemos construir uma GSRN equivalente

- A ideia é, cada vez que um símbolo nulificável ocorre do lado direito de uma regra, acrescentar uma regra nova à GLC original sem a ocorrência deste símbolo

- Por exemplo, na gramática anterior havia a regra $S \rightarrow AB$
  - A esta regra serão acrescentadas as regras
    - $S \rightarrow A$ (substituindo B pela cadeia vazia)
    - $S \rightarrow B$ (substituindo A pela cadeia vazia)

(continua)

- Evitamos acrescentar a regra $S \to \varepsilon$ (substituindo tanto A como B pela cadeia vazia) porque queremos gerar uma GSRN

- Essa ideia simples não funciona caso a cadeia vazia seja gerada pela gramática, isto é, se S é um símbolo nulificável
  - Neste caso, devemos acrescentar um novo símbolo inicial que gere a cadeia vazia

Exemplo

- Considere a gramática
  - S → AB
  - A → a | ε
  - B → b

- Primeiro, identificamos os símbolos nulificáveis
  - A (é nulificável porque A → ε)

- Aplicamos o algoritmo para construir a GSRN equivalente:
  - Para a regra S → AB :
    - Acrescentamos S → A (substituindo B pela cadeia vazia)
    - Acrescentamos S → B (substituindo A pela cadeia vazia)

(continua)

$A \to a \mid \varepsilon$

ou

$A \to a$

$A \to \varepsilon$

- Para a regra $A \to a$:
  - Como ela não contém ε diretamente, essa regra não precisa ser modificada no contexto de eliminação de regras nulas
- Para a regra $B \to b$:
  - Não precisamos adicionar mais regras, pois B não é nulificável
- Regras resultantes na GSRN equivalente:
  - $S \to AB \mid A \mid B$
  - $A \to a$
  - $B \to b$