

Linguagens Formais e Autômatos

Linguagem Sensível ao Contexto

Eduardo Furlan Miranda

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

Linguagem Sensível ao Contexto (LSC)

- Algumas abreviações
 - LC Livre de Contexto
 - LLC Linguagem Livre de Contexto
 - GLC Gramática Livre de Contexto
 - RLC Regra Livre de Contexto
 - SC Sensível ao Contexto
 - LSC Linguagem Sensível ao Contexto
 - GSC Gramática Sensível ao Contexto
 - RSC Regra Sensível ao Contexto
 - LR Linguagem Regular
 - GR Gramática Regular
 - MT Máquina de Turing
 - AF Autômato Finito

Hierarquia de Chomsky

3/20

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

apenas estas



Gramáticas	Regras	Ex. de ling. geradas, e reconhecedor
GR (tipo 3) Regulares	$A \rightarrow aB, A \rightarrow b, (A \rightarrow \varepsilon \text{ apenas para o símbolo inicial, se permitido})$ $A, B \in V$ (variáveis) $a, b \in T$ (terminais)	<ul style="list-style-type: none"> $\{ \varepsilon, b, ab, aab, aaab, \dots \}$ $= \{ a^n b \mid n \geq 0 \} \cup \{ \varepsilon \}$ Autômato finito
GLC (tipo 2) Livres de Contexto	$A \rightarrow \alpha$ $A \in V, \alpha \in (V \cup T)^*$ A: 1 única variável	<ul style="list-style-type: none"> $\{ ab, aabb, aaabbb, aaaabbbb, \dots \}$ $= \{ a^n b^n \mid n > 0 \}$ Autômato com pilha
GSC (tipo 1) Sensíveis ao Contexto	$\alpha \rightarrow \beta$ $\alpha \in (V \cup T)^+, \beta \in (V \cup T)^*$ $ \alpha \leq \beta $ $S \rightarrow \varepsilon$, se S não aparece do lado direito de nenhuma regra	<ul style="list-style-type: none"> $\{ abc, aabbcc, aaabbbccc, \dots \}$ $= \{ a^n b^n c^n \mid n > 0 \}$ Autômato linearmente limitado
GI (tipo 0) Irrestrita ou geral	$\alpha \rightarrow \beta$ $\alpha, \beta \in (V \cup T)^*$ α : pelo menos 1 símbolo de V	<ul style="list-style-type: none"> $\{ a, aa, aaaa, aaaaaaaa, \dots \}$ $= \{ a^{2^n} \mid n \geq 0 \}$ Máquina de Turing

Linguagens Livres de Contexto (LLC)

(já visto)

- Regras de produção na forma $A \rightarrow \alpha$
 - Geradas por GLC com regras de produção que dependem apenas do símbolo atual, e não do “contexto em que ele aparece”
 - Depende da sequência de símbolos que precede ou segue
- GLC possuem regras na forma $A \rightarrow \alpha$, onde $\alpha \in (V \cup T)^*$ e $A \in V$

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}	$a, aa, aaaa$

Tipos de Gramáticas	Regras	Exemplos de linguagens geradas
Livres de Contexto GLC (tipo 2)	$A \rightarrow \alpha$ $A \in V$, $\alpha \in (V \cup T)^*$ A: 1 única variável esq.	$\{ ab, aabb, aaabbb, aaaabbbb, \dots \}$ $= \{ a^n b^n \mid n > 0 \}$

Exemplo

- Preciso reconhecer o comando **print(texto)**
- A gramática pode ser definida como:
 - $S \rightarrow \text{"print" } E$
 - $E \rightarrow \text{"(" } T \text{ ")"}$
 - $T \rightarrow \text{"texto"}$

" \rightarrow " = símbolo de produção ou regra de derivação
" \Rightarrow " = símbolo de derivação
- Exemplo de reconhecimento:
 - $S \Rightarrow \text{"print" } E \Rightarrow \text{"print" "(" } T \text{ ")" } \Rightarrow \text{"print" "(" "texto" ")"}$
 - Para que "print(texto)" seja reconhecido, precisa existir uma sequência de derivações começando de S até essa cadeia, seguindo as regras da gramática definida
 - "print" e "texto" seriam tokens reconhecidos na etapa de análise léxica

Linguagens Sensíveis ao Contexto (LSC)

RSC = Regra Sensível ao Contexto

- Regras de produção na forma $\alpha A \beta \rightarrow \alpha \gamma \beta$ ($\gamma \neq \epsilon$)
 - Geradas por gramáticas que têm regras de produção que “dependem do contexto em que o símbolo atual aparece”
 - Dependem não apenas do símbolo atual, mas também da sequência de símbolos que o precedem ou o seguem
 - α e β representam o contexto
Ex.: $aAc \rightarrow aBc$ só pode ser aplicada se a A estiver entre a e c
- RSC : o tamanho do lado esquerdo é \leq ao do lado direito
- GSC : é tupla $G = (V, T, P, S)$ onde toda regra em P é uma RSC

$|\alpha|$ representa o tamanho (ou comprimento) da cadeia α

Sensíveis ao Contexto
GSC (tipo 1)

$\alpha \rightarrow \beta$
 $\alpha \in (V \cup T)^+$, $\beta \in (V \cup T)^*$
 $|\alpha| \leq |\beta|$
 $S \rightarrow \epsilon$, se S não aparece do lado dir.

$\{ abc, aabbcc, aaabbbccc, \dots \}$
 $= \{ a^n b^n c^n \mid n > 0 \}$

Compiladores e analisadores sintáticos

- Na prática, as GSC não são utilizadas na construção de analisadores sintáticos, pois acarretaria em um mecanismo de análise complicado e ineficiente
 - A derivação de uma cadeia em uma GSC não possui uma estrutura que represente tão bem a cadeia do ponto de vista hierárquico,
 - como é o caso das árvores de análise para as LLC
- Compiladores usam analisadores guiados por GLC ,
 - e alguns mecanismos mais sofisticados que as GSC , para dar conta de análise de contexto

Gramática Sensível ao Contexto (GSC)

- Seja a gramática $G = (V, T, P, S)$. As RSC têm as formas:
 - $\alpha \rightarrow \beta$, com $\alpha, \beta \in (V \cup T)^+$, α possuindo pelo menos uma ocorrência de variável, e $|\alpha| \leq |\beta|$
 - Tanto α quanto β são sequências não vazias de símbolos, onde cada símbolo pode ser uma variável V ou um terminal T
 - $S \rightarrow \varepsilon$, se S é o símbolo inicial e não ocorre à direita de todas as regras da gramática
 - Esta é a condição adicional que permite que o símbolo inicial S seja substituído pelo símbolo vazio ε , desde que S não ocorra à direita de nenhuma regra da gramática

Linguagem Sensível ao Contexto (LSC)

- L é uma LSC se, e somente se, existe uma GSC G tal que $L = L(G)$
 - LSC = Linguagem Sensível ao Contexto
 - GSC = Gramática Sensível ao Contexto (regras da forma $\alpha A \beta \rightarrow \alpha \gamma \beta$, com $\gamma \neq \epsilon$, exceto para $S \rightarrow \epsilon$)
 - $L(G)$ = Conjunto de cadeias geradas por G a partir do símbolo inicial
- LSCs são definidas por gramáticas com regras sensíveis ao contexto ($\alpha A \beta \rightarrow \alpha \gamma \beta$), onde a substituição de A por γ depende do contexto (α e β)
- Em contraste, LLCs são definidas por gramáticas com regras livres de contexto ($A \rightarrow \gamma$), sem dependência do contexto

Exemplo - LSC

- Seja a linguagem $\{a^n b^n c^n \mid 0 < n\}$ gerada pela GSC:

- $S \rightarrow ABc$ ← inicial

- $AB \rightarrow AABBC$

- $CB \rightarrow BC$

- $Cc \rightarrow cc$

- $Bc \rightarrow bc$

- $Bb \rightarrow bb$

- $Ab \rightarrow ab$

- $Aa \rightarrow aa$

Derivação de $a^3b^3c^3$:

$S \Rightarrow ABc \Rightarrow AABBCc$

$AABBCc \Rightarrow AAABBCBCc$

$AAABBCBCc \Rightarrow AAABBBCCc$

$AAABBBCCc \Rightarrow AAABBBCCc$

$AAABBBCCc \Rightarrow AAABBBcc$

$AAABBBcc \Rightarrow AAABbbcc$

$AAABbbcc \Rightarrow (\dots) \Rightarrow Aaabbbcc$

$Aaabbbcc \Rightarrow \boxed{aaabbbcc}$

- Na derivação de **aaabbbccc** , as regras poderiam ter sido aplicadas em outra ordem, que não a exibida, e assim mesmo **a mesma cadeia seria derivada**
- Por exemplo, a partir de AAABBCBCc , podemos derivar **AAABBCBcc** ao invés de AAABBBCCc
 - a partir de **AAABBCBcc** só poderíamos derivar AAABBBCCc,
 - enquanto que a partir da cadeia AAABBBCCc só é possível derivar AAABBBCCc
- Desta forma, podemos observar que **qualquer sequência de aplicações de regras** da gramática **gera cadeias da linguagem** em questão

Exemplo - GSC

GSC que gera a $L = \{ ww \mid w \in \{ a, b \}^* \}$:

1) $S \rightarrow RT \mid aa \mid bb$  $S = \text{inicial}$

2) $R \rightarrow RaA \mid RbB \mid aaM \mid abN \mid baO \mid bbP$

3) $Aa \rightarrow aA$

$Ab \rightarrow bA$

$AT \rightarrow Ta$

4) $Ba \rightarrow aB$

$Bb \rightarrow bB$

$BT \rightarrow Tb$

5) $Ma \rightarrow aM$

$Mb \rightarrow bM$

$MT \rightarrow aa$

6) $Na \rightarrow aN$

$Nb \rightarrow bN$

$NT \rightarrow ab$

7) $Oa \rightarrow aO$

$Ob \rightarrow bO$

$OT \rightarrow ba$

8) $Pa \rightarrow aP$

$Pb \rightarrow bP$

$PT \rightarrow bb$

(continua)

Exemplo - GSC (continuação)

- A gramática é projetada para processar palavras **w** de forma ordenada e sensível ao contexto
- Ela lê **w** no prefixo **R** e garante, por meio de regras intermediárias e sensíveis ao contexto, que o sufixo **T** será uma cópia idêntica de **w**, resultando em **ww**
- Exemplo de derivação

$S \Rightarrow RT \Rightarrow RaAT \Rightarrow bbPaAT \Rightarrow bbaPAT \Rightarrow bbaPTa \Rightarrow bba bba$

$R \rightarrow RaA$

$R \rightarrow bbP$

$Pa \rightarrow aP$

$AT \rightarrow Ta$

$PT \rightarrow bb$

Gramática Sensível ao Contexto (GSC)

- GSC, como possuem regras apenas na forma

$$\alpha \rightarrow \beta, \text{ onde } |\alpha| \leq |\beta| \quad ,$$

garantem que é computável o problema de determinar se uma cadeia pertence ou não à linguagem gerada pela gramática G

- Isto é, existe um programa de computador que implementa esta decisão
- Isto é consequência do fato de que a cada uso de regra no processo de verificação, a cadeia sendo gerada nunca diminui de tamanho

GSC

- Devido às regras das GSC, onde uma cadeia gerada nunca diminui de tamanho,
 - é possível criar um programa de computador que decida se uma cadeia pertence à linguagem gerada por uma gramática G
- Com um número finito de aplicações de regras que não aumentam o tamanho da cadeia, podemos concluir que a cadeia w pertence à linguagem $L(G)$ e é derivada pela gramática
- Porém, se todas as possíveis derivações resultam em cadeias maiores que w , isso indica que G não pode derivar w

RLC é RSC

- Com a exceção de regras da forma $A \rightarrow \varepsilon$, toda RLC é uma RSC
- Toda GLC que não gera a cadeia vazia pode ser convertida em uma Gramática Livre de Contexto Sem Regras Nulas (GSRN) equivalente
- Toda LLC é também uma LSC, uma vez que a classe das linguagens sensíveis ao contexto inclui todas as linguagens livres de contexto

- Existem LSC que não são LLC
- Ex.: $L_{abc} = \{ a^n b^n c^n \mid 0 < n \}$
 - Existe a necessidade de "manter a contagem" simultânea de três símbolos diferentes (a, b e c)
 - As GLCs têm uma capacidade limitada de "memória", pois só podem substituir um não terminal por vez, independentemente do contexto
 - Para garantir a igualdade entre as quantidades de a, b e c, seria necessário um mecanismo que as GLCs não possuem

LSCs são fechadas sob união ($L_1 \cup L_2$)

- Gramáticas Originais
 - Sejam $L_1 = L(G_1)$ e $L_2 = L(G_2)$, onde G_1 e G_2 são GSCs
- Renomeação de Variáveis
 - Garantir que os não-terminais de G_1 e G_2 não se sobreponham
- Nova Gramática G
 - Incluir todas as regras de G_1 e G_2
 - Adicionar nova regra: $S \rightarrow S_1 \mid S_2$ (sendo S o novo símbolo inicial)
- Validação
 - A regra $S \rightarrow S_1 \mid S_2$ é livre de contexto, mas válida em GSCs
- Resultado
 - $L(G) = L_1 \cup L_2$, provando o fechamento

LSCs são fechadas sob concatenação (L_1L_2)

- Combinação das Gramáticas
 - Mesmo processo de renomeação de não-terminais
- Regra de Concatenação
 - Adicionar $S \rightarrow S_1S_2$ à gramática combinada
- Tipo da Regra
 - Livre de contexto (caso especial de GSC com $\alpha = \beta = \varepsilon$)
- Geração da Linguagem
 - $L(G) = \{uv \mid u \in L_1, v \in L_2\}$
- Conclusão
 - L_1L_2 é uma LSC

LSCs são fechadas sob estrela $(L_1)^*$

- Caso Base ($\varepsilon \in L_1^*$)
 - Adicionar $S \rightarrow \varepsilon$ (se necessário; S não aparece do lado direito em outras regras)
- Regra de Repetição
 - Incluir $S \rightarrow S_1S$ para gerar cadeias de tamanho ≥ 1
- Alternativa (se $\varepsilon \notin L_1$)
 - Usar $S \rightarrow S_1S \mid S_1$ para L_1^+
- Validação
 - Todas as regras da nova GSC G^* devem satisfazer a condição de crescimento ou preservação do tamanho ($|\alpha| \leq |\beta|$)
- Resultado Final
 - Como L_1^* é gerada por uma GSC válida (G^*), L_1^* é uma LSC