ACH2043
INTRODUÇÃO À TEORIA DA COMPUTAÇÃO

2. Linguagens Livres-do-Contexto

Referência:
SIPSER, M. Introdução à Teoria da Computação.
2ª edição, Ed. Thomson

Prof. Marcelo S. Lauretto
marcelolauretto@usp.br
www.each.usp.br/lauretto

2.1 Gramáticas livres-do-contexto

- No Capítulo 1 estudamos dois métodos formais para descrever linguagens regulares:
  - Autômatos finitos
  - Expressões regulares
- Foi demonstrado que algumas linguagens, mesmo simples, não são regulares
  - P.ex. $\{0^n 1^n \mid n \geq 0\}$
- Neste capítulo, apresentamos as **gramáticas livres-do-contexto (GLC)**, um método mais poderoso de descrever linguagens
  - Podem descrever linguagens com certas estruturas recursivas

2.1 Gramáticas livres-do-contexto

- Inicialmente usadas no estudo de linguagem natural
  - Estruturas de uma frase são recursivas
- Aplicação importante: análise sintática em linguagens de programação
  - Estruturas recursivas, p.ex
    - Expressões aritméticas: $2 \times [3 \div (1 - 4 + 2) + 2 \times 3]$
    - Estruturas de desvio, repetição, etc, que induzem blocos aninhados
- Coleção de linguagens associadas com gramáticas livres de contexto são denominadas **linguagens livres-do-contexto**
  - Incluem as linguagens regulares

2.1 Gramáticas livres-do-contexto

- Exemplo de uma gramática livre-do-contexto (denotada por G_1):

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$

- Uma gramática consiste de uma coleção de **regras de substituição** (ou de **produção**)
  - Cada regra aparece em uma linha da gramática na forma

$$V \longrightarrow w$$

    na qual:
    - V é uma **variável**
    - w é uma cadeia composta por variáveis e outros símbolos chamados **terminais**

2.1 Gramáticas livres-do-contexto

- Exemplo de uma gramática livre-do-contexto (denotada por G_1): (cont)

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$

  - Variáveis são usualmente representadas por letras maiúsculas (ao menos nos exemplos simples aqui apresentados)
  - Os terminais são análogos ao alfabeto de entrada e são usualmente representados por letras minúsculas, números ou símbolos especiais
  - Uma das variáveis é designada a **variável inicial**
    - Usualmente a que aparece no lado esquerdo da 1ª regra
- Identifique as regras, variáveis e terminais na gramática G_1

2.1 Gramáticas livres-do-contexto

- Uma gramática permite gerar cada cadeia de uma linguagem da seguinte maneira:
  1. Escreva a variável inicial
  2. Para cada variável V escrita na cadeia, encontre uma regra contendo V no lado esquerdo $(V \longrightarrow w)$
     Substitua V pelo lado direito daquela regra
     - Ex: se a cadeia atual é

       $$aVcSb$$

       e existe uma regra $V \longrightarrow bSa$, pode-se substituir V por bSa na cadeia acima, resultando em

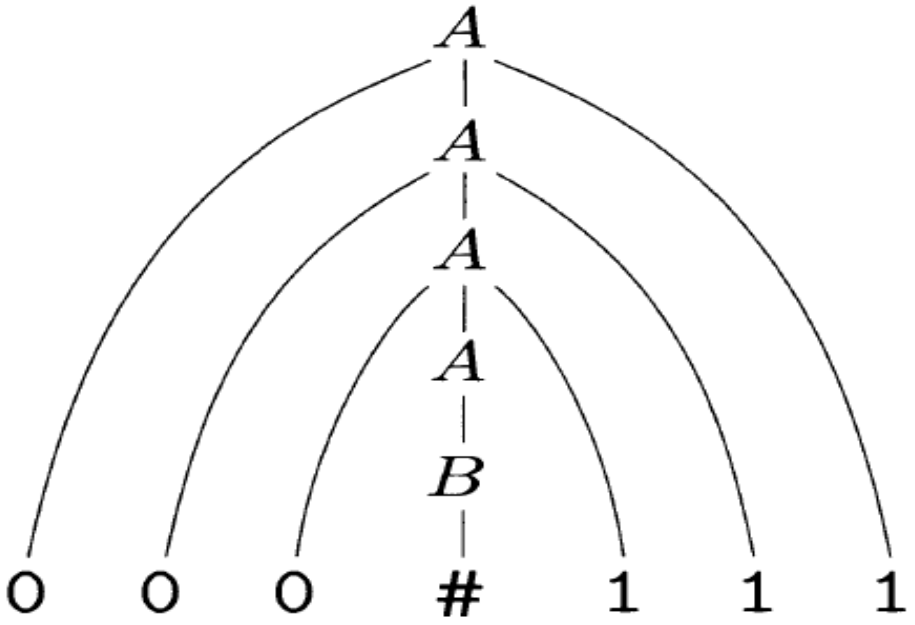
       $$a\underline{bSa}cSb$$
  3. Repita o passo 2 até não haver mais nenhuma variável

2.1 Gramáticas livres-do-contexto

- Ex: a gramática G_1 gera a cadeia $000\#111$
  - Sequência de substituições para obter uma cadeia é denominada **derivação**
  - A derivação da cadeia $000\#111$ na gramática G_1 é

$$A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 000A111 \Rightarrow 000B111 \Rightarrow 000\#111$$

  - A derivação também pode ser representada por uma **árvore sintática**:



2.1 Gramáticas livres-do-contexto

- Todas as cadeias geradas dessa forma constituem a **linguagem da gramática**
  - Notação: $L(G)$
- Qual a linguagem da gramática G_1 do exemplo anterior?

- Toda cadeia que pode ser gerada por uma gramática livre de contexto é denominada **linguagem livre-do-contexto (LLC)**

2.1 Gramáticas livres-do-contexto

- Gramática abaixo, denotada por G_2, descreve um fragmento da estrutura sintática do Inglês

$$
\begin{aligned}
\langle \text{SENTENCE} \rangle &\rightarrow \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
\langle \text{NOUN-PHRASE} \rangle &\rightarrow \langle \text{CMPLX-NOUN} \rangle \mid \langle \text{CMPLX-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{VERB-PHRASE} \rangle &\rightarrow \langle \text{CMPLX-VERB} \rangle \mid \langle \text{CMPLX-VERB} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{PREP-PHRASE} \rangle &\rightarrow \langle \text{PREP} \rangle \langle \text{CMPLX-NOUN} \rangle \\
\langle \text{CMPLX-NOUN} \rangle &\rightarrow \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \\
\langle \text{CMPLX-VERB} \rangle &\rightarrow \langle \text{VERB} \rangle \mid \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \\
\langle \text{ARTICLE} \rangle &\rightarrow \texttt{a} \mid \texttt{the} \\
\langle \text{NOUN} \rangle &\rightarrow \texttt{boy} \mid \texttt{girl} \mid \texttt{flower} \\
\langle \text{VERB} \rangle &\rightarrow \texttt{touches} \mid \texttt{likes} \mid \texttt{sees} \\
\langle \text{PREP} \rangle &\rightarrow \texttt{with}
\end{aligned}
$$

2.1 Gramáticas livres-do-contexto

- Exemplos de cadeias geradas:

```
a boy sees
the boy sees a flower
a girl with a flower likes the boy
```

  – Derivação da 1ª cadeia:

$$
\begin{aligned}
\langle \text{SENTENCE} \rangle &\Rightarrow \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \langle \text{CMPLX-NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \texttt{a}\ \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \texttt{a boy}\ \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \texttt{a boy}\ \langle \text{CMPLX-VERB} \rangle \\
&\Rightarrow \texttt{a boy}\ \langle \text{VERB} \rangle \\
&\Rightarrow \texttt{a boy sees}
\end{aligned}
$$

2.1 Gramáticas livres-do-contexto
Definição formal

- Definição 2.2:

A ***context-free grammar*** is a 4-tuple (V, Σ, R, S), where

1. V is a finite set called the ***variables***,
2. Σ is a finite set, disjoint from V, called the ***terminals***,
3. R is a finite set of ***rules***, with each rule being a variable and a string of variables and terminals, and
4. $S \in V$ is the start variable.

2.1 Gramáticas livres-do-contexto
Definição formal

- Frequentemente especificamos uma gramática escrevendo apenas suas regras
  - Variáveis são identificadas pelos símbolos que aparecem no lado esquerdo das regras
  - Terminais são todos os demais símbolos
  - Por convenção, a variável inicial é a que aparece no lado esquerdo da 1ª regra

2.1 Gramáticas livres-do-contexto
Definição formal

- Se u, v, w são cadeias de variáveis e terminais, e $A \longrightarrow v$ é uma regra da gramática, dizemos que uAw **origina** uvw, denotado por $uAw \Longrightarrow uvw$.
  - Pode-se também dizer que uvw é uma **derivação direta** de uAw
- Dizemos que u **deriva** v, escrito $u \Longrightarrow^* v$, se $u = v$ ou se existe uma sequência $u_1, u_2, \ldots, u_k$ para $k \geq 0$ tal que

$$u_1 \Longrightarrow u_2 \Longrightarrow \cdots \Longrightarrow u_k$$

- A **linguagem de uma gramática** G é o conjunto de todas as cadeias de terminais derivadas de sua variável inicial S, ou seja,

$$L(G) = \{w \in \Sigma^* \mid S \Longrightarrow^* w\}$$

2.1 Gramáticas livres-do-contexto
Exemplos

- Exemplo 2.3 (modificado):
  - Considere a gramática $G_3 = (\{S\}, \{0,1,(,)\}, S, R)$. O conjunto de regras, *R*, é:

$$S \longrightarrow (S)|SS|0|1|\varepsilon$$

  - Alguns exemplos de cadeias geradas: ε, 0, 1, (000), (1(0)), (()1(01)), etc
  - $L(G_3) =$?

2.1 Gramáticas livres-do-contexto
Exemplos

- Exemplo 2.4:

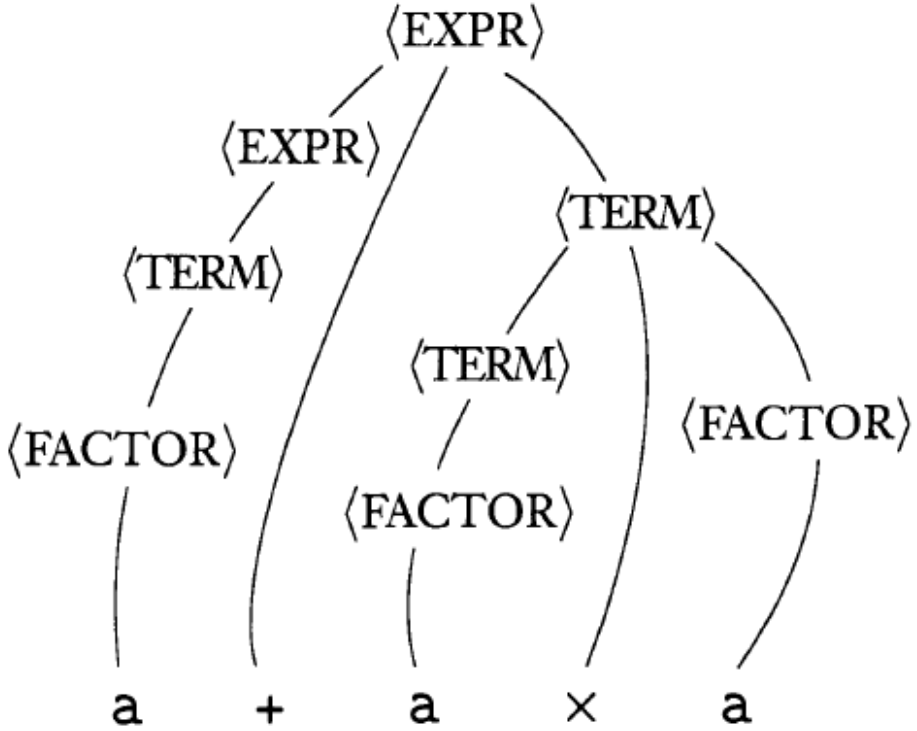
Consider grammar $G_4 = (V, \Sigma, R, \langle \text{EXPR} \rangle)$.
V is $\{\langle \text{EXPR} \rangle, \langle \text{TERM} \rangle, \langle \text{FACTOR} \rangle\}$ and Σ is $\{\texttt{a}, \texttt{+}, \texttt{x}, \texttt{(}, \texttt{)}\}$. The rules are

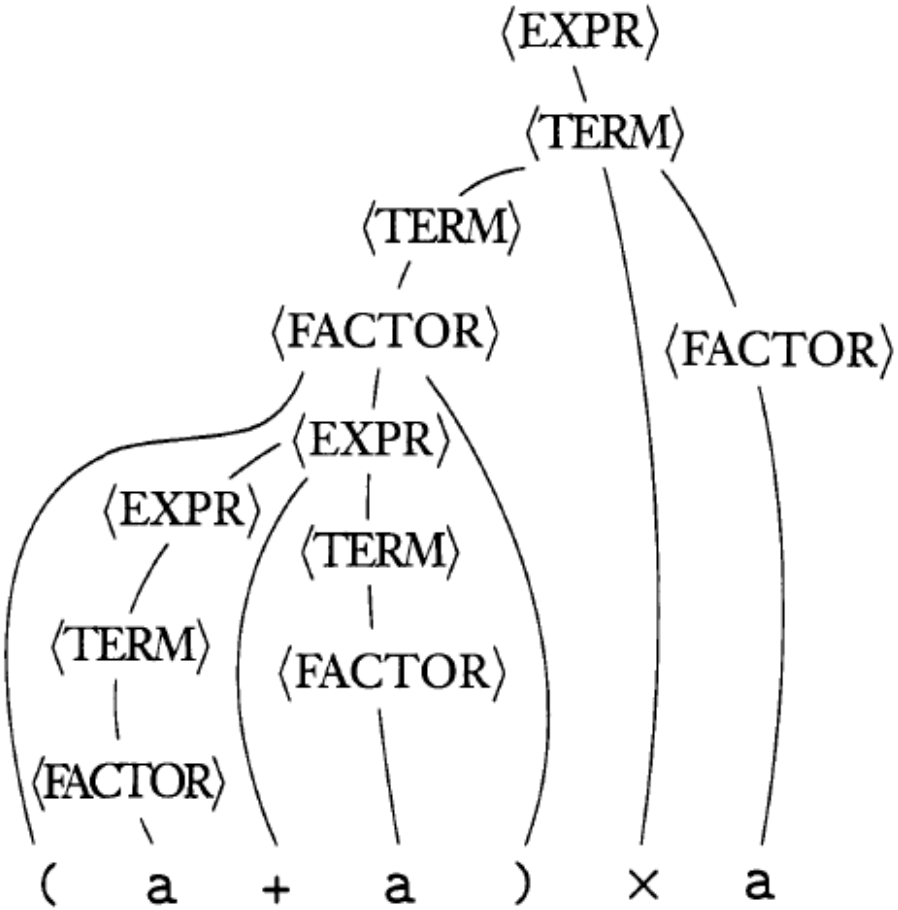
$$\begin{aligned} \langle \text{EXPR} \rangle &\rightarrow \langle \text{EXPR} \rangle \texttt{+} \langle \text{TERM} \rangle \mid \langle \text{TERM} \rangle \\ \langle \text{TERM} \rangle &\rightarrow \langle \text{TERM} \rangle \texttt{x} \langle \text{FACTOR} \rangle \mid \langle \text{FACTOR} \rangle \\ \langle \text{FACTOR} \rangle &\rightarrow \texttt{(} \langle \text{EXPR} \rangle \texttt{)} \mid \texttt{a} \end{aligned}$$

  - As cadeias $a + a \times a$ e $(a + a) \times a$ podem ser geradas com a gramática G_4.
    - Árvores sintáticas são apresentadas no próximo slide

2.1 Gramáticas livres-do-contexto
Exemplos

- Exemplo 2.4 (cont):





2.1 Gramáticas livres-do-contexto
Projetando GLCs

- Projeto de uma GLC requer certo grau de criatividade
- Todavia, alguns princípios podem ser considerados

1. Linguagens livres-do-contexto (LLCs) são, usualmente, composições de LLCs mais simples
   – Para projetar uma GLC G para uma linguagem que representa a união entre duas LLCs L_1 e L_2, construa primeiramente as gramáticas
   G_1 para L_1 e G_2 para L_2; em seguida, combine as regras de G_1 e de G_2, e adicione uma nova regra

$$S \longrightarrow S_1 | S_2 \ ,$$

   onde
   - S, S_1 e S_2 são as variáveis iniciais de G, G_1 e G_2, respectivamente

2.1 Gramáticas livres-do-contexto
Projetando GLCs

1. Linguagens livres-do-contexto (LLCs) são, usualmente, composições de LLCs mais simples (cont)
   – Por exemplo, para obter uma gramática para a linguagem
   $$\{0^n 1^n | n \geq 0\} \cup \{1^n 0^n | n \geq 0\},$$
   primeiro construímos duas gramáticas mais simples:

$$\{0^n 1^n | \, n \geq 0\} \qquad S_1 \rightarrow 0S_1 1 \mid \varepsilon$$

$$\{1^n 0^n | \, n \geq 0\} \qquad S_2 \rightarrow 1S_2 0 \mid \varepsilon$$

   Em seguida adicionamos a regra $S \longrightarrow S_1 | S_2$, resultando na gramática final

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow 0S_1 1 \mid \varepsilon \\ S_2 &\rightarrow 1S_2 0 \mid \varepsilon \end{aligned}$$

2.1 Gramáticas livres-do-contexto
Projetando GLCs

2. Projetar uma GLC para uma linguagem regular é simples, se conseguirmos projetar o AFD ou AFN para aquela linguagem
   – Ver apresentação “Introdução às Gramáticas”, onde apresentamos a equivalência entre autômatos finitos e gramáticas lineares à direita (que são um caso particulares de GLCs)

2.1 Gramáticas livres-do-contexto
Projetando GLCs

3. Certas LLCs contêm cadeias com duas subcadeias que são “ligadas” no sentido de que uma máquina para uma linguagem como essa precisaria memorizar uma quantidade de informação ilimitada sobre uma das subcadeias para saber se ela corresponde apropriadamente à outra subcadeia
   - Essa situação ocorre na linguagem $\{0^n 1^n | n \geq 0\}$, pois uma máquina precisaria memorizar o número de 0s de modo a verificar que ele é igual ao número de 1s
   - Pode-se construir uma GLC para lidar com essa situação usando uma regra na forma $R \longrightarrow uRv$, que gera cadeias nas quais a parte contendo os u’s corresponde às partes contendo os v’s

2.1 Gramáticas livres-do-contexto
Projetando GLCs

4. Em linguagens mais complexas, as cadeias podem conter certas estruturas que aparecem recursivamente como parte de outras estruturas (ou delas mesmas)
   - Essa situação ocorre na gramática que gera expressões aritméticas no Exemplo 2.4
     - Sempre que o símbolo a aparece, em vez dele uma expressão parentizada pode aparecer recursivamente
   - Para atingir esse efeito, coloque o símbolo da variável que gera a estrutura na posição das regras correspondente a onde aquela estrutura pode aparecer recursivamente

2.1 Gramáticas livres-do-contexto
Projetando GLCs

- Exercício 2.4:

Give context-free grammars that generate the following languages. alphabet Σ is $\{0,1\}$.

[A] **a.** $\{w \mid w$ contains at least three 1s$\}$

b. $\{w \mid w$ starts and ends with the same symbol$\}$

c. $\{w \mid$ the length of w is odd$\}$

[A] **d.** $\{w \mid$ the length of w is odd and its middle symbol is a 0$\}$

e. $\{w \mid w = w^{\mathcal{R}}$, that is, w is a palindrome$\}$

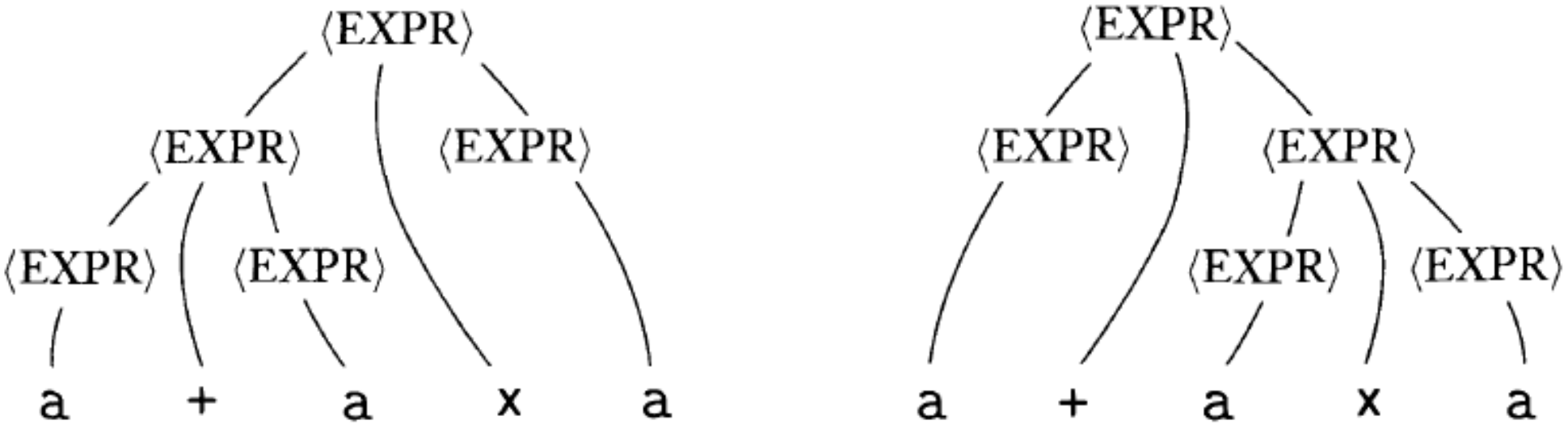
f. The empty set

2.1 Gramáticas livres-do-contexto
Ambiguidade

- Às vezes uma gramática pode gerar a mesma cadeia de várias maneiras diferentes
- Tal cadeia terá várias árvores sintáticas diferentes e, portanto, vários significados diferentes
- Se uma gramática gera a mesma cadeia de várias maneiras diferentes, dizemos que a cadeia é **derivada ambiguamente** nessa gramática
- Se uma cadeia gera alguma cadeia ambiguamente, dizemos que a gramática é **ambígua**
- Ambiguidade pode ser indesejável em certas aplicações
  - P.ex. em linguagens de programação: um programa deve ter uma única interpretação

2.1 Gramáticas livres-do-contexto
Ambiguidade

- Exemplo: considere a gramática G_5:
  - $\langle \text{EXPR} \rangle \rightarrow \langle \text{EXPR} \rangle + \langle \text{EXPR} \rangle \mid \langle \text{EXPR} \rangle \times \langle \text{EXPR} \rangle \mid (\langle \text{EXPR} \rangle) \mid \texttt{a}$
  - Essa gramática gera a cadeia $a + a \times a$ ambiguamente:



  - Qual seria o valor dessa expressão sob cada árvore sintática se $a = 3$?
    - à esquerda: 18; à direita: 12

2.1 Gramáticas livres-do-contexto
Ambiguidade

- Exemplo: considere a gramática G_5 (cont):

$$\langle \text{EXPR} \rangle \rightarrow \langle \text{EXPR} \rangle + \langle \text{EXPR} \rangle \mid \langle \text{EXPR} \rangle \times \langle \text{EXPR} \rangle \mid (\langle \text{EXPR} \rangle) \mid \texttt{a}$$

  - G_5 não captura as relações de precedência usuais e, portanto, pode agrupar o $+$ antes do $\times$ e vice-versa
  - Por outro lado, a gramática G_4 gera exatamente a mesma linguagem, mas toda cadeia gerada tem uma árvore sintática única
  - Logo, G_4 é não-ambígua, enquanto G_5 é ambígua

- A gramática G_2, que descreve um fragmento da estrutura sintática do Inglês, também é ambígua
  - Sentença “the girl touches the boy with the flower” tem duas derivações diferentes (e duas interpretações diferentes!)

2.1 Gramáticas livres-do-contexto
Ambiguidade

- Atenção: quando dizemos que uma gramática gera uma cadeia ambiguamente, queremos dizer que a cadeia tem duas árvores sintáticas diferentes, e não duas derivações diferentes
  - Duas derivações podem diferir meramente pela ordem na qual elas substituem variáveis e ainda assim não na sua estrutura geral
  - Para nos concentrarmos na estrutura da gramática, definimos um tipo de derivação que substitui variáveis em uma ordem fixa

2.1 Gramáticas livres-do-contexto
Ambiguidade

- Uma **derivação mais à esquerda** de uma cadeia *w* é aquela em que, a cada passo, a variável remanescente escolhida para ser substituída é a variável mais à esquerda na forma sentencial intermediária
  - Ex: derivação de $(a + a) \times a$ pela gramática G_4:
    $\langle E \rangle \Rightarrow \langle T \rangle \Rightarrow \langle T \rangle \times \langle F \rangle \Rightarrow \langle F \rangle \times \langle F \rangle \Rightarrow (\langle E \rangle) \times \langle F \rangle$
    $\Rightarrow (\langle E \rangle + \langle T \rangle) \times \langle F \rangle \Rightarrow (\langle T \rangle + \langle T \rangle) \times \langle F \rangle \Rightarrow (\langle F \rangle + \langle T \rangle) \times \langle F \rangle$
    $\Rightarrow (a + \langle T \rangle) \times \langle F \rangle \Rightarrow (a + \langle F \rangle) \times \langle F \rangle \Rightarrow (a + a) \times \langle F \rangle \Rightarrow (a + a) \times a$

2.1 Gramáticas livres-do-contexto
Ambiguidade

- Definição 2.7:

  A string w is derived ***ambiguously*** in context-free grammar G if it has two or more different leftmost derivations. Grammar G is ***ambiguous*** if it generates some string ambiguously.

  – Em certas situações, quando temos uma gramática ambígua podemos encontrar uma gramática não-ambígua que gera a mesma linguagem

  – Algumas linguagens livres-do-contexto, entretanto, podem ser geradas apenas por gramáticas ambíguas

  – Tais linguagens são chamadas **inerentemente ambíguas**

  – Ex:

$$\{a^i b^j c^k | i = j \text{ ou } j = k\}$$

  - Não há gramática que tenha derivação única para a cadeia aaabbbccc, por exemplo

2.1 Gramáticas livres-do-contexto
Ambiguidade

- Exemplo em linguagens de programação:
  - Suponha que uma gramática contenha a regra:

    Statement → **if** Condition **then** Statement |
    if Condition **then** Statement **else** Statement |
    ...

    Condition → ...
  - Algumas estruturas ambíguas podem aparecer:
    - A sentença

      if a **then if** b **then** s **else** s2

      pode ser interpretada como

      if a **then** { **if** b **then** s } **else** s2

      ou

      if a **then** { **if** b **then** s **else** s2 }

      dependendo da associação do **else** com o 1º ou com o 2º **if**

2.1 Gramáticas livres-do-contexto
Ambiguidade

- Exemplo em linguagens de programação:
  - Suponha que uma gramática contenha a regra:

    Statement → **if** Condition **then** Statement |
    if Condition **then** Statement **else** Statement |
    ...
    Condition → ...
  - Formas de resolver ambiguidade:
    - Modificação da gramática para torná-la não-ambígua
      - p.ex tornar **endif / fi** obrigatório
        Statement → **if** Condition **then** Statement **fi** |
        if Condition **then** Statement **else** Statement **fi** |
        ...
    - Manter a gramática ambígua, mas tornar a sentença sensível ao contexto, associando o **else** com o último **if** encontrado

2.1 Gramáticas livres-do-contexto
Forma Normal de Chomsky

- Quando se trabalha com gramáticas livres-do-contexto, é frequentemente tê-las em forma simplificada
- Uma das formas mais simples e úteis é chamada **forma normal de Chomsky (FNC)**
  - Útil para projetar algoritmos para análise sintática

2.1 Gramáticas livres-do-contexto
Forma Normal de Chomsky

- **Definição 2.8:**

  Uma GLC está na Forma Normal de Chomsky se:

  a) Toda regra de substituição é da forma

     $A \rightarrow BC$ ou $A \rightarrow a$

     onde B,C são variáveis, a é símbolo terminal;

  b) A variável inicial S não pode aparecer no lado direito de nenhuma regra;

  c) Somente a variável inicial pode ter a regra

     $S \rightarrow \varepsilon$.

2.1 Gramáticas livres-do-contexto
Forma Normal de Chomsky

- Conversão de uma GLC G = (V, Σ, R, S) para FNC:

a) Adicionar nova variável inicial S_0 e adicionar a regra $S_0 \rightarrow S$;

b) Eliminação de regras $A \rightarrow \varepsilon$

b_1) Remover a regra;

b_2) Para toda regra $R \rightarrow u\ A\ v$, adicionar $R \rightarrow u\ v$;

Nota: Fazer isso para cada ocorrência de A.

Ex: se $R \rightarrow u\ A\ v\ A\ w$, deve-se acrescentar 3 regras:

$$R \rightarrow u\ v\ A\ w, \quad R \rightarrow u\ A\ v\ w, \quad R \rightarrow u\ v\ w$$

b_3) Se tivermos a regra $R \rightarrow A$ e se $R \rightarrow \varepsilon$ não tiver sido previamente eliminado, adicionar $R \rightarrow \varepsilon$

(posteriormente, essa regra também será removida se $R \neq S_0$)

b_4) Repetir até eliminar todas as ocorrências.

2.1 Gramáticas livres-do-contexto
Forma Normal de Chomsky

- Conversão de uma GLC G = (V, Σ, R, S) para FNC (cont.):

  c) Remoção de regras unitárias A → B:

  c1) Remover a regra;

  c2) Para toda regra B → u, acrescentamos A → u, a menos que essa seja uma regra unitária já removida.

  c3) Repetir para todas as regras unitárias.

2.1 Gramáticas livres-do-contexto
Forma Normal de Chomsky

- Conversão de uma GLC G = (V, Σ, R, S) para FNC (cont.):

  d) Converter todas as regras remanescentes para a forma apropriada A → BC ou A → a:

  d1) Se A → $u_1u_2 \ldots u_k$, onde $k \geq 3$ e u_i é variável ou símbolo terminal, então substituir esta regra por:

  $$A \rightarrow u_1A_1,\ A_1 \rightarrow u_2A_2,\ A_2 \rightarrow u_3A_3, \ldots, A_{k-2} \rightarrow u_{k-1}u_k.$$

  d2) Se k=2, então substituir qualquer terminal u_i na(s) regra(s) precedente(s) por uma nova variável U_i, e adicionar a regra $U_i \rightarrow u_i$.

2.1 Gramáticas livres-do-contexto
Forma Normal de Chomsky

- Exemplo 2.10: conversão da GLC abaixo para a FNC:

  1. GLC original G_6 é mostrada à esquerda
  Resultado da introdução da nova variável inicial é apresentado à direita

$$S \to ASA \mid \mathtt{a}B$$
$$A \to B \mid S$$
$$B \to \mathtt{b} \mid \varepsilon$$

$$\boldsymbol{S_0 \to S}$$
$$S \to ASA \mid \mathtt{a}B$$
$$A \to B \mid S$$
$$B \to \mathtt{b} \mid \varepsilon$$

  2. Remoção de regras vazias $B \longrightarrow \varepsilon$ (esq) e $A \longrightarrow \varepsilon$ (dir)

$$S_0 \to S$$
$$S \to ASA \mid \mathtt{a}B \mid \mathbf{a}$$
$$A \to B \mid S \mid \boldsymbol{\varepsilon}$$
$$B \to \mathtt{b} \mid \varepsilon$$

$$S_0 \to S$$
$$S \to ASA \mid \mathtt{a}B \mid \mathtt{a} \mid \boldsymbol{SA} \mid \boldsymbol{AS} \mid \boldsymbol{S}$$
$$A \to B \mid S \mid \varepsilon$$
$$B \to \mathtt{b}$$

2.1 Gramáticas livres-do-contexto
Forma Normal de Chomsky

- Exemplo 2.10: conversão da GLC abaixo para a FNC:

3a. Remoção de regras unitárias $S \longrightarrow S$ (esq) e $S_0 \longrightarrow S$ (dir)

$$\begin{aligned} S_0 &\to S \\ S &\to ASA \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\ A &\to B \mid S \\ B &\to \mathtt{b} \end{aligned}$$

$$\begin{aligned} S_0 &\to \boldsymbol{ASA} \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\ S &\to ASA \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\ A &\to B \mid S \\ B &\to \mathtt{b} \end{aligned}$$

3b. Remoção de regras unitárias $A \longrightarrow B$ (esq) e $A \longrightarrow S$ (dir)

$$\begin{aligned} S_0 &\to ASA \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\ S &\to ASA \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\ A &\to B \mid S \mid \mathtt{b} \\ B &\to \mathtt{b} \end{aligned}$$

$$\begin{aligned} S_0 &\to ASA \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\ S &\to ASA \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\ A &\to S \mid \mathtt{b} \mid \boldsymbol{ASA} \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\ B &\to \mathtt{b} \end{aligned}$$

2.1 Gramáticas livres-do-contexto
Forma Normal de Chomsky

- Exemplo 2.10: conversão da GLC abaixo para a FNC:

  4. Conversão das regras remanescentes para a forma apropriada acrescentando variáveis e regras adicionais

$$
\begin{aligned}
S_0 &\to AA_1 \mid UB \mid \mathtt{a} \mid SA \mid AS \\
S &\to AA_1 \mid UB \mid \mathtt{a} \mid SA \mid AS \\
A &\to \mathtt{b} \mid AA_1 \mid UB \mid \mathtt{a} \mid SA \mid AS \\
A_1 &\to SA \\
U &\to \mathtt{a} \\
B &\to \mathtt{b}
\end{aligned}
$$

Algoritmos para Processamento de Gramáticas Livres do Contexto

- Problema:
  - Dado uma GLC G e uma cadeia w, decidir se G gera a cadeia w
  - Redefinição do problema em termos de linguagem:
    - A_{GLC} = { (H, x) | H é uma GLC que gera a cadeia x }
    - Dada uma Problema: (G, W) $\in$ A_{GLC}?

7.2 Algoritmos para Processamento de Gramáticas Livres do Contexto

- Se G for uma gramática qualquer:
  - Testar todas as possíveis derivações: loop infinito
- Se G estiver na Forma Normal de Chomsky:
  - Teorema: qualquer derivação de uma cadeia não vazia *w* tem exatamente 2*n*-1 passos, onde *n* = |*w*|.
    - Ideia da demonstração:
      - n-1 passos para gerar a cadeia com n variáveis
      - n passos para substituir as variáveis por terminais
    - Método de força bruta (para n>0): custo exponencial
      - Teste todas as derivações possíveis com 2n-1 passos.
      - Se alguma das derivações gerar w, aceite; senão, rejeite

7.2 Algoritmos para Processamento de Gramáticas Livres do Contexto

- Algoritmo CYK (Cocke–Younger–Kasami)
  - Polinomial (se G estiver na forma normal de Chomsky)
- **Programação dinâmica**: uso de soluções de subproblemas menores para resolver subproblemas maiores (até chegar à solução do problema original)
- Tabela n×n:
  - Para i ≤ j, a (i, j) entrada da tabela contém todas as varíaveis que geram a subcadeia $w_i\, w_{i+1} \ldots w_j$
  - Tratam-se subcadeias de tamanhos crescentes (começando de 1)

7.2 Algoritmo CYK – programação dinâmica

$D =$ "On input $w = w_1 \cdots w_n$:

1. If $w = \varepsilon$ and $S \to \varepsilon$ is a rule, *accept*. [[handle $w = \varepsilon$ case]]
2. For $i = 1$ to n: [[examine each substring of length 1]]
3. For each variable A:
4. Test whether $A \to \mathtt{b}$ is a rule, where $\mathtt{b} = w_i$.
5. If so, place A in $table(i, i)$.
6. For $l = 2$ to n: [[l is the length of the substring]]
7. For $i = 1$ to $n - l + 1$: [[i is the start position of the substring]]
8. Let $j = i + l - 1$, [[j is the end position of the substring]]
9. For $k = i$ to $j - 1$: [[k is the split position]]
10. For each rule $A \to BC$:
11. If $table(i, k)$ contains B and $table(k + 1, j)$ contains C, put A in $table(i, j)$.
12. If S is in $table(1, n)$, *accept*. Otherwise, *reject*."

7.2 Algoritmo CYK – programação dinâmica

Exemplo: Conversão de uma GLC para FNC:

Gramática original:

$S \rightarrow aSb \mid bSa \mid SS \mid \varepsilon$

1) Criação de nova variável inicial:

$S_0 \rightarrow S$

$S \rightarrow aSb \mid bSa \mid SS \mid \varepsilon$

2) Eliminação de substituições ε :

$S_0 \rightarrow S \mid \varepsilon$

$S \rightarrow aSb \mid bSa \mid SS \mid ab \mid ba \mid S$

3) Eliminação de regras unitárias:

$S_0 \rightarrow \varepsilon \mid aSb \mid bSa \mid SS \mid ab \mid ba$

$S \rightarrow aSb \mid bSa \mid SS \mid ab \mid ba$

4) Padronização:

$S_0 \rightarrow \varepsilon \mid AT \mid BU \mid SS \mid AB \mid BA$

$S \rightarrow AT \mid BU \mid SS \mid AB \mid BA$

$T \rightarrow SB$

$U \rightarrow SA$

$A \rightarrow a$

$B \rightarrow b$

7.2 Algoritmo CYK – programação dinâmica

Exemplo (cont):

Aplicação algoritmo CYK:

Grámática na FNC:

$S_0 \rightarrow \varepsilon \mid AT \mid BU \mid SS \mid AB \mid BA$

$S \rightarrow AT \mid BU \mid SS \mid AB \mid BA$

$T \rightarrow SB$

$U \rightarrow SA$

$A \rightarrow a$

$B \rightarrow b$

Cadeia:

a b a a b b

Tabela:

	a	b	a	a	b	b
a	A	S_0, S	U	∅	∅	S_0, S
b		B	S_0, S	U	S_0, S	T
a			A	∅	∅	S_0, S
a				A	S_0, S	T
b					B	∅
b						B