SBVLIFA: Linguagens Formais e Autômatos

Aula 07: Gramáticas e Linguagens Livres de Contexto

Bacharelado em Ciência da Computação
Prof. Dr. David Buzatto

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Campus São João da Boa Vista

Linguagens Livres de Contexto

- Linguagens Livres de Contexto

Tipo	Classe de Linguagens	Modelo de Gramática	Modelo de Reconhecedor
0	Recursivamente enumeráveis	Irrestrita	Máquina de Turing
1	Sensíveis ao contexto	Sensível ao contexto	Máquina de Turing com fita limitada
2	**Livres de contexto**	**Livre de contexto**	**Autômato de pilha**
3	Regulares	Linear (direita ou esquerda)	Autômato finito

Linguagens Livres de Contexto

- A classe das linguagens livres de contexto é uma classe de linguagens maior que a das linguagens regulares;
- A notação natural e recursiva para a representação de linguagens livres de contexto é a **Gramática Livre de Contexto (CFG)**;
- As CFGs desempenham papel central na tecnologia de compiladores e interpretadores, sendo aplicadas na construção dos analisadores sintáticos (*parsers*), na definição de linguagens específicas de domínio e em diversos outros cenários;
- Os autômatos de pilha (com pilha/à pilha) possuem uma notação semelhante à dos autômatos finitos e são equivalentes às gramáticas livres de contexto, pois são também capazes de representar, como um dispositivo reconhecedor, as linguagens livres de contexto.

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Um Exemplo Informal:**
  - A linguagem dos palíndromos: um palíndromo é uma string lida da mesma forma da esquerda para a direita e vice-versa. A string w é um palíndromo se e somente se $w = w^R$. Focaremos na linguagem dos palíndromos sobre o alfabeto $\{0, 1\}$, tendo como exemplos 0110, 11011, ε, mas não 011 ou 0101. Aplicando o lema do bombeamento em L_{pal} verifica-se que ela não é uma linguagem regular, pois se o fosse, tomando a constante associada n e considerando $w = 0^n10^n$, poderíamos desmembrar w em $w = xyz$ com y sendo um ou mais 0's do primeiro grupo, pois $|xy| \leq n$ e $y \neq \varepsilon$. Desse modo, xz, que também teria que estar em L_{pal} se L_{pal} fosse regular, teria menos 0's à esquerda do único 1 do que à direita do 1, fazendo com que xz não seja um palíndromo. Intuitivamente, é possível perceber que para que uma string de L_{pal} seja reconhecida, é necessário se ter uma "memória" do que aconteceu previamente.

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Um Exemplo Informal:**
  - Perceba que que há uma definição natural e recursiva de quando uma string de 0's e 1's está em L_{pal}. Como base, sabemos que algumas strings óbvias estão em L_{pal} e depois exploramos a ideia de que se uma string é um palíndromo ela deve começar e terminar com o mesmo símbolo. Além disso, se removermos o primeiro e o último símbolo, a string ainda é um palíndromo:
  - **Base:** ε, 0 e 1 são palíndromos;
  - **Indução:** Se w é um palíndromo, então $0w0$ e $1w1$ também são palíndromos. Nenhuma string é um palíndromo de 0's e 1's, a menos que ela decorra dessa base e dessa regra de indução.
  - Uma CFG é uma notação formal para expressar tais definições recursivas de linguagens e ela consiste em uma ou mais variáveis que representam classes de strings, ou seja, linguagens.

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Um Exemplo Informal:**
  - As regras que definem os palíndromos, expressas na notação de gramática livre de contexto, podem ser vistas abaixo:

1. $P \rightarrow \varepsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Um Exemplo Informal:**
  - As regras que definem os palíndromos, expressas na notação de gramática livre de contexto, podem ser vistas abaixo:

Variável

Conjunto de 5 regras

1. $P \rightarrow \varepsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Um Exemplo Informal:**
  - As regras que definem os palíndromos, expressas na notação de gramática livre de contexto, podem ser vistas abaixo:

1. $P \rightarrow \varepsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

Base: inclui as strings ε, 0 e 1; Nenhum dos termos da direita contém uma variável.

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Um Exemplo Informal:**
  - As regras que definem os palíndromos, expressas na notação de gramática livre de contexto, podem ser vistas abaixo:

1. $P \rightarrow \varepsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

Indução: se w é da classe P, então $0w0$ e $1w1$ também o são; Os termos da direita contém uma variável.

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Definição Formal:**
  1. Existe um conjunto finito de símbolos que formam as strings da linguagem, chamado de alfabeto de terminais ou símbolos terminais: $\{0, 1\}$

1. $P \rightarrow \varepsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Definição Formal:**

2. Existe um conjunto finito de variáveis, também chamadas de não terminais ou categorias sintáticas. Cada variável representa uma linguagem, ou seja, um conjunto de strings: $\{P\}$

1. $P \rightarrow \varepsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

Linguagens Livres de Contexto

Gramáticas Livres de Contexto

- **Definição Formal:**

3. Uma das variáveis representa a linguagem que está sendo definida, sendo chamada de símbolo de início: P

Neste exemplo há somente uma variável, sendo assim, é ela que é o símbolo inicial.

1. $P \rightarrow \varepsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

Linguagens Livres de Contexto

Gramáticas Livres de Contexto

- **Definição Formal:**

4. Existe um conjunto finito de produções ou regras que representam a definição recursiva de uma linguagem.

Produção ou regra

1. $P \rightarrow \varepsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Definição Formal:**

4. Existe um conjunto finito de produções ou regras que representam a definição recursiva de uma linguagem.

Cabeça (*head*) da produção:
definição parcial de uma variável

1. $P \rightarrow \varepsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

Linguagens Livres de Contexto

Gramáticas Livres de Contexto

- **Definição Formal:**

4. Existe um conjunto finito de produções ou regras que representam a definição recursiva de uma linguagem.

Símbolo de produção:
lido como "leva à" ou "implica em"

1. $P \rightarrow \varepsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

Linguagens Livres de Contexto

Gramáticas Livres de Contexto

- **Definição Formal:**

4. Existe um conjunto finito de produções ou regras que representam a definição recursiva de uma linguagem.

1. $P \rightarrow \varepsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

Corpo da produção: uma string de zero ou mais terminais e/ou variáveis que representa um modo de formar strings na linguagem da variável da cabeça

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Definição Formal:**

$$G = (V, T, P, S)$$

- G: gramática livre de contexto, uma quádrupla, onde:
  - V: conjunto finito de variáveis;
  - T ou Σ: conjunto finito, disjunto de V, de símbolos terminais;
  - P: conjunto finito de produções;
  - S: $S \in V$ é a variável ou símbolo inicial.

Linguagens Livres de Contexto

Gramáticas Livres de Contexto

- **Exemplo:** a linguagem de expressões aritméticas com os operadores + e $*$, e identificadores na forma $(a+b)(a+b+0+1)^*$:

- $G = (V, T, P, S)$, onde:
  - $V = \{E, I\}$
  - $T = \{+,\ *,\ (,\ ),\ a,\ b,\ 0,\ 1\}$
  - $P =$
  - $S = E$

1. $E \to I$
2. $E \to E + E$
3. $E \to E * E$
4. $E \to (E)$
5. $I \to a$
6. $I \to b$
7. $I \to Ia$
8. $I \to Ib$
9. $I \to I0$
10. $I \to I1$

Linguagens Livres de Contexto

Gramáticas Livres de Contexto

- **Exemplo:** a linguagem de expressões aritméticas com os operadores + e $*$, e identificadores na forma $(a+b)(a+b+0+1)^*$:

- $G = (V, T, P, S)$, onde:
  - $V = \{E, I\}$
  - $T = \{+,\ *,\ (,\ ),\ a,\ b,\ 0,\ 1\}$
  - $P =$
  - $S = E$

1. $E \to I$
2. $E \to E + E$
3. $E \to E * E$
4. $E \to (E)$
5. $I \to a$
6. $I \to b$
7. $I \to Ia$
8. $I \to Ib$
9. $I \to I0$
10. $I \to I1$

notação compacta

1. $E \to I \mid E + E \mid E * E \mid (E)$
2. $I \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

Linguagens Livres de Contexto

Gramáticas Livres de Contexto

- **Inferências sobre pertinência de strings em linguagens:**
  - **Inferência Recursiva:** utilização das regras do corpo para a cabeça.

	String inferida	Para linguagem de	Produção usada	String(s) usada(s)
(i)	a	I	5	–
(ii)	b	I	6	–
(iii)	$b0$	I	9	(ii)
(iv)	$b00$	I	9	(iii)
(v)	a	E	1	(i)
(vi)	$b00$	E	1	(iv)
(vii)	$a + b00$	E	2	$(v), (vi)$
$(viii)$	$(a + b00)$	E	4	(vii)
(ix)	$a * (a + b00)$	E	3	$(v), (viii)$

1. $E \to I$
2. $E \to E + E$
3. $E \to E * E$
4. $E \to (E)$

5. $I \to a$
6. $I \to b$
7. $I \to Ia$
8. $I \to Ib$
9. $I \to I0$
10. $I \to I1$

Linguagens Livres de Contexto

Gramáticas Livres de Contexto

- **Inferências sobre pertinência de strings em linguagens:**
  - **Derivação:** utilização das regras da cabeça para o corpo.
    - **Base:** para qualquer string α de terminais e variáveis, dizemos que $\alpha \overset{*}{\underset{G}{\Rightarrow}} \alpha$, isto é, qualquer string deriva de si própria.
    - **Indução:** se $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$ e $\beta \underset{G}{\Rightarrow} \gamma$, então $\alpha \overset{*}{\underset{G}{\Rightarrow}} \gamma$. Isto é, se α pode se tornar β por meio de zero ou mais etapas e, se mais uma etapa nos leva de β para γ, então α pode se tornar γ. Em outras palavras, $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$ significa que existe uma sequência de strings $\gamma_1, \gamma_2, \ldots, \gamma_n$, para algum $n \geq 1$, tal que:
      1. $\alpha = \gamma_1$,
      2. $\beta = \gamma_n$, e
      3. Para $i = 1, 2, \ldots, n-1$, temos $\gamma_i \Rightarrow \gamma_{i+1}$

Se G é subentendida, então usamos $\overset{*}{\Rightarrow}$ no lugar de $\overset{*}{\underset{G}{\Rightarrow}}$

  - **Exemplo:** inferir que $a * (a + b00)$ está em E:

$$E \Rightarrow E * E \Rightarrow I * E \Rightarrow a * E \Rightarrow a * (E) \Rightarrow a * (E + E) \Rightarrow a * (I + E) \Rightarrow$$
$$a * (a + E) \Rightarrow a * (a + I) \Rightarrow a * (a + I0) \Rightarrow a * (a + I00) \Rightarrow a * (a + b00)$$

1. $E \to I$
2. $E \to E + E$
3. $E \to E * E$
4. $E \to (E)$
5. $I \to a$
6. $I \to b$
7. $I \to Ia$
8. $I \to Ib$
9. $I \to I0$
10. $I \to I1$

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Derivações mais à esquerda e mais à direita:**
  - **Derivação mais à esquerda (*lm/leftmost*):** exigir que em cada etapa, a variável mais à esquerda seja substituída por um de seus corpos. Indicada usando as relações $\underset{lm}{\Rightarrow}$ e $\overset{*}{\underset{lm}{\Rightarrow}}$ para uma ou mais etapas, respectivamente;
  - **Exemplo:** $E \underset{lm}{\Rightarrow} E * E \underset{lm}{\Rightarrow} I * E \underset{lm}{\Rightarrow} a * E \underset{lm}{\Rightarrow} a * (E) \underset{lm}{\Rightarrow} a * (E + E) \underset{lm}{\Rightarrow} a * (I + E) \underset{lm}{\Rightarrow}$
    $a * (a + E) \underset{lm}{\Rightarrow} a * (a + I) \underset{lm}{\Rightarrow} a * (a + I0) \underset{lm}{\Rightarrow} a * (a + I00) \underset{lm}{\Rightarrow} a * (a + b00)$
  - **Derivação mais à direita (*rm/rightmost*):** exigir que em cada etapa, a variável mais à direita seja substituída por um de seus corpos. Indicada usando as relações $\underset{rm}{\Rightarrow}$ e $\overset{*}{\underset{rm}{\Rightarrow}}$ para uma ou mais etapas, respectivamente;
  - **Exemplo:** $E \underset{rm}{\Rightarrow} E * E \underset{rm}{\Rightarrow} E * (E) \underset{rm}{\Rightarrow} E * (E + E) \underset{rm}{\Rightarrow} E * (E + I) \underset{rm}{\Rightarrow} E * (E + I0) \underset{rm}{\Rightarrow}$
    $E * (E + I00) \underset{rm}{\Rightarrow} E * (E + b00) \underset{rm}{\Rightarrow} E * (I + b00) \underset{rm}{\Rightarrow} E * (a + b00) \underset{rm}{\Rightarrow}$
    $I * (a + b00) \underset{rm}{\Rightarrow} a * (a + b00)$

1. $E \to I$
2. $E \to E + E$
3. $E \to E * E$
4. $E \to (E)$
5. $I \to a$
6. $I \to b$
7. $I \to Ia$
8. $I \to Ib$
9. $I \to I0$
10. $I \to I1$

Linguagens Livres de Contexto

Gramáticas Livres de Contexto

- **Derivações mais à esquer**
  - **Derivação mais à esquerda (*lm/leftmost*):** exigir que em cada etapa, el mais à esquerda seja
    usando as relações $\underset{lm}{\Rightarrow}$ e $\overset{*}{\underset{lm}{\Rightarrow}}$ para uma ou mais etapas,
    vamente;
  - **Exemplo:** $E \underset{lm}{\Rightarrow} E * E \underset{lm}{\Rightarrow} I * E \underset{lm}{\Rightarrow} a * E \underset{lm}{\Rightarrow} a * (E) \underset{lm}{\Rightarrow} a * (E + E) \underset{lm}{\Rightarrow} a * (I + E) \underset{lm}{\Rightarrow}$
    $a * (a + E) \underset{lm}{\Rightarrow} a * (a + I) \underset{lm}{\Rightarrow} a * (a + I0) \underset{lm}{\Rightarrow} a * (a + I00) \underset{lm}{\Rightarrow} a * (a + b00)$
  - **Derivação mais à direita (*rm/rightmost*):** exigir que em cada etapa, el mais à direita seja substituída por um de seus corpos.
    usando as relações $\underset{rm}{\Rightarrow}$ e $\overset{*}{\underset{rm}{\Rightarrow}}$ para uma ou mais etapas,
    vamente;
  - **Exemplo:** $E \underset{rm}{\Rightarrow} E * E \underset{rm}{\Rightarrow} E * (E) \underset{rm}{\Rightarrow} E * (E + E) \underset{rm}{\Rightarrow} E * (E + I) \underset{rm}{\Rightarrow} E * (E + I0) \underset{rm}{\Rightarrow}$
    $E * (E + I00) \underset{rm}{\Rightarrow} E * (E + b00) \underset{rm}{\Rightarrow} E * (I + b00) \underset{rm}{\Rightarrow} E * (a + b00) \underset{rm}{\Rightarrow}$
    $I * (a + b00) \underset{rm}{\Rightarrow} a * (a + b00)$

Se w é uma string de terminais e A é uma variável, então $A \overset{*}{\Rightarrow} w$ se e somente se $A \overset{*}{\underset{lm}{\Rightarrow}} w$, e $A \overset{*}{\Rightarrow} w$ se e somente se $A \overset{*}{\underset{rm}{\Rightarrow}} w$. Ou seja, qualquer derivação tem uma derivação equivalente mais à esquerda e uma derivação equivalente mais à direita.

Pode-se concluir que: $E \overset{*}{\underset{lm}{\Rightarrow}} a * (a + b00)$

Pode-se concluir que: $E \overset{*}{\underset{rm}{\Rightarrow}} a * (a + b00)$

$+ E$

$* E$

4. $E \rightarrow (E)$
5. $I \rightarrow a$
6. $I \rightarrow b$
7. $I \rightarrow Ia$
8. $I \rightarrow Ib$
9. $I \rightarrow I0$
10. $I \rightarrow I1$

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **A linguagem de uma gramática:**
  - Se $G = (V, T, P, S)$ é uma CFG, a linguagem de G, denotada por $L(G)$, é o conjunto de strings terminais que têm derivações desde o símbolo de início, ou seja:

$$L(G) = \{w \text{ em } T^* \mid S \overset{*}{\underset{G}{\Rightarrow}} w\}$$

  - Se uma linguagem L é uma linguagem de alguma CFG, então L é dita uma **Linguagem Livre de Contexto (CFL)**.

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

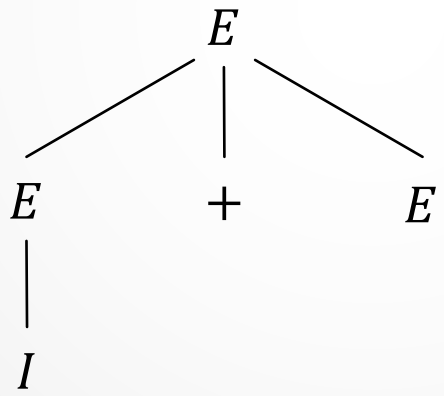
- **Formas sentenciais:**
  - As formas sentenciais são as derivações que produzem strings a partir do símbolo de início;
  - Se $G = (V, T, P, S)$ é uma CFG, então qualquer string α em $(V \cup T)^*$ tal que $S \overset{*}{\Rightarrow} \alpha$ é uma forma sentencial;
  - Se $S \underset{lm}{\overset{*}{\Rightarrow}} \alpha$, então α é uma forma sentencial à esquerda;
  - Se $S \underset{rm}{\overset{*}{\Rightarrow}} \alpha$, então α é uma forma sentencial à direita;
  - A linguagem $L(G)$ é constituída pelas formas sentenciais que estão em T^*, ou seja, as formas sentenciais constituídas de apenas terminais.

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Árvores de análise sintática:**
  - Para $G = (V, T, P, S)$, as árvores de análise sintática para G são árvores com as seguintes condições:
    1. Cada nó interior, ou seja, um nó que tem pelo menos um filho, é rotulado por uma variável em V;
    2. Cada nó folha é rotulado por uma variável, um terminal ou ε. Se o nó folha for rotulado por ε, ele deve ser o único filho de seu pai;
    3. Se um nó interior é rotulado por A e seus filhos são rotulados por $X_1, X_2, \ldots, X_k$, respectivamente, a partir da esquerda, então $A \rightarrow X_1, X_2, \ldots, X_k$ é uma produção em P.

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Árvores de análise sintática:**
  - **Exemplo 1:** a árvore de análise sintática para a derivação de $I + E$ a partir de E:

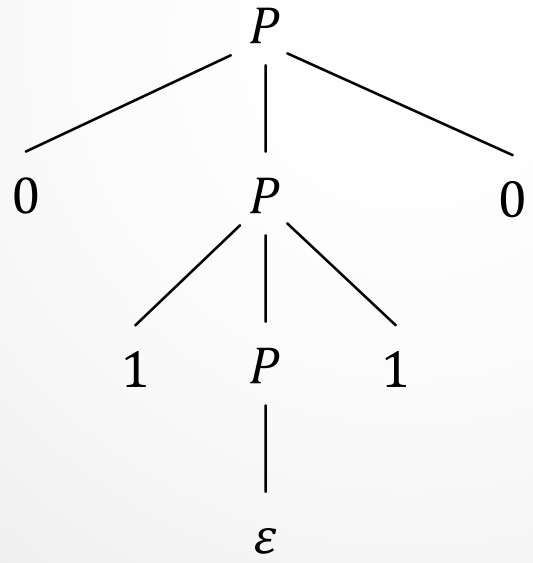


1. $E \rightarrow I$
2. $E \rightarrow E + E$
3. $E \rightarrow E * E$
4. $E \rightarrow (E)$
5. $I \rightarrow a$
6. $I \rightarrow b$
7. $I \rightarrow Ia$
8. $I \rightarrow Ib$
9. $I \rightarrow I0$
10. $I \rightarrow I1$

Linguagens Livres de Contexto

Gramáticas Livres de Contexto

- **Árvores de análise sintática:**
  - **Exemplo 2:** a árvore de análise sintática para a derivação de 0110 a partir de P:

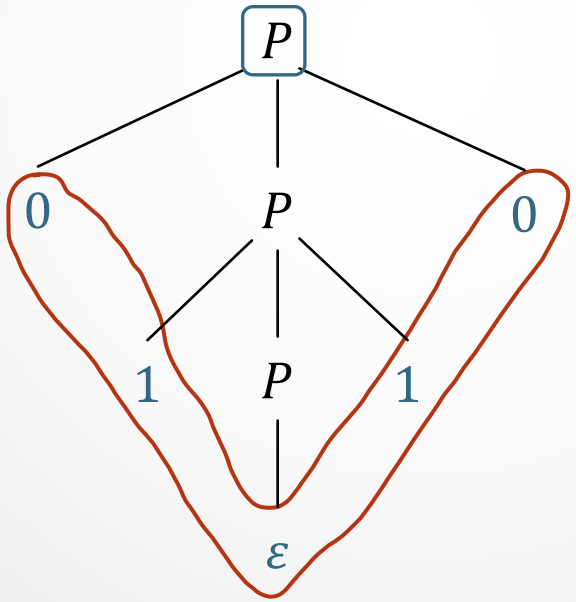


1. $P \to \varepsilon$
2. $P \to 0$
3. $P \to 1$
4. $P \to 0P0$
5. $P \to 1P1$

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Árvores de análise sintática:**
  - **Exemplo 2:** a árvore de análise sintática para a derivação de 0110 a partir de P:

Raiz rotulada pelo símbolo de início

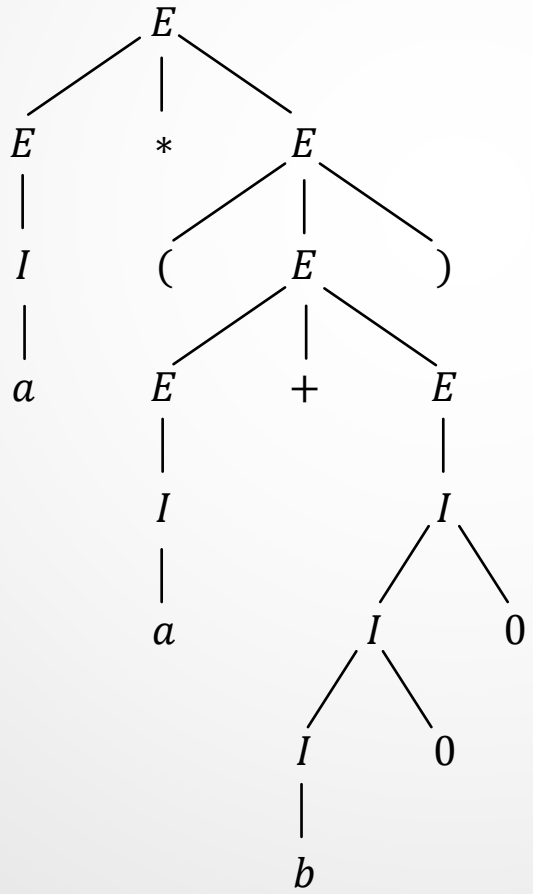


1. $P \to \varepsilon$
2. $P \to 0$
3. $P \to 1$
4. $P \to 0P0$
5. $P \to 1P1$

String de terminais em que todas as folhas são rotuladas por um terminal ou ε

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

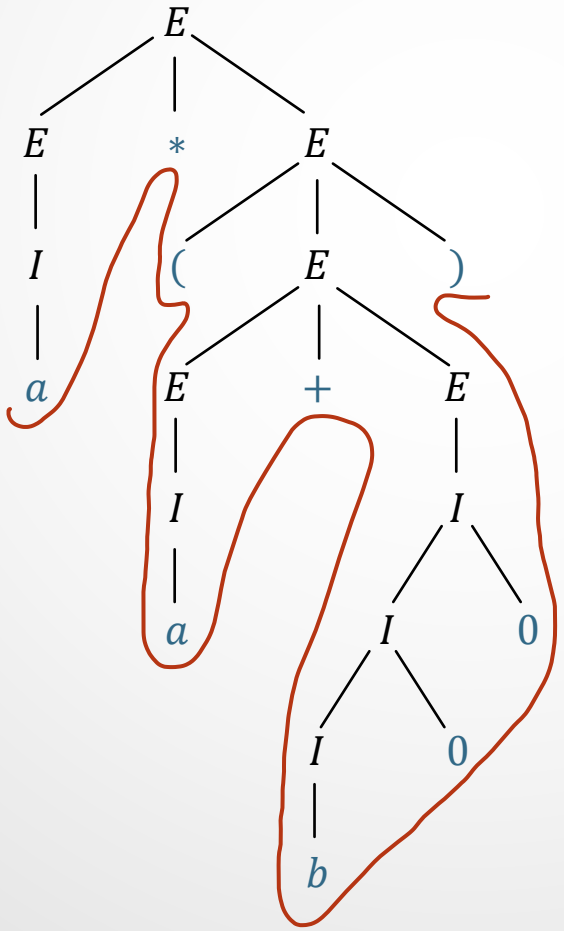
- **Árvores de análise sintática:**
  - **Exemplo 3:** a árvore de análise sintática para a derivação de $a * (a + b00)$ a partir de E:



1. $E \rightarrow I$
2. $E \rightarrow E + E$
3. $E \rightarrow E * E$
4. $E \rightarrow (E)$
5. $I \rightarrow a$
6. $I \rightarrow b$
7. $I \rightarrow Ia$
8. $I \rightarrow Ib$
9. $I \rightarrow I0$
10. $I \rightarrow I1$

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Árvores de análise sintática:**
  - **Exemplo 3:** a árvore de análise sintática para a derivação de $a * (a + b00)$ a partir de E:



1. $E \rightarrow I$
2. $E \rightarrow E + E$
3. $E \rightarrow E * E$
4. $E \rightarrow (E)$
5. $I \rightarrow a$
6. $I \rightarrow b$
7. $I \rightarrow Ia$
8. $I \rightarrow Ib$
9. $I \rightarrow I0$
10. $I \rightarrow I1$

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

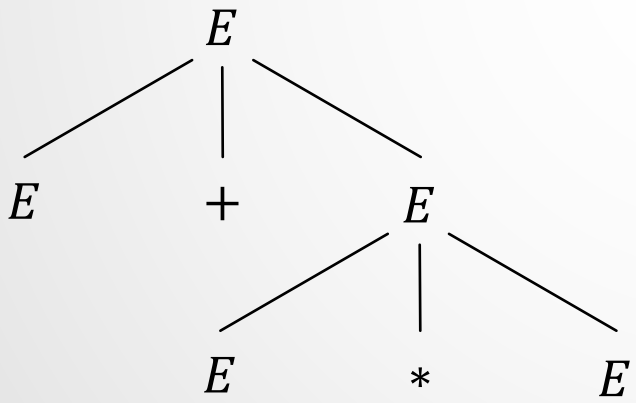
- **Inferência, derivações e árvores de análise sintática:**
  - Dada uma gramática $G = (V, T, P, S)$, os seguintes itens são equivalentes:
    1. O procedimento de inferência recursiva determina que a string de terminais w está na linguagem da variável A;
    2. $A \overset{*}{\Rightarrow} w$;
    3. $A \underset{lm}{\overset{*}{\Rightarrow}} w$;
    4. $A \underset{rm}{\overset{*}{\Rightarrow}} w$;
    5. Existe uma árvore de análise sintática com raiz A e resultado w.
  - As provas formais para as equivalências entre os itens apresentados acima podem ser verificadas na obra Hopcroft et al. (2003).

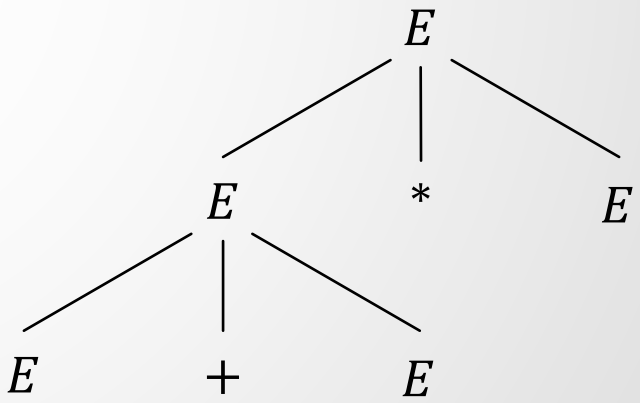
Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Ambiguidade em gramáticas e linguagens:**
  - Uma gramática é ambígua quando, para uma mesma string da linguagem da gramática, **existe mais de uma árvore de análise sintática**. Há como reprojetar a gramática de modo a remover tal ambiguidade, entretanto, em alguns casos, quando a CFL é inerentemente ambígua, isso não é possível;
  - Múltiplas derivações para uma mesma string não implicam, obrigatoriamente, em ambiguidade, entretanto para gramáticas não-ambíguas as derivações mais à esquerda e mais à direita são únicas;
  - **Gramáticas ambíguas:** a gramática da linguagem de expressões aritméticas com os operadores + e $*$ e identificadores, nos permite gerar expressões com qualquer sequência de operadores + e $*$, e as produções $E \rightarrow E + E \mid E * E$ não estabelece nenhuma ordem particular. Lembre-se que na aritmética a operação de multiplicação deve ser realizada antes da operação de adição caso ambas estejam em um mesmo nível da expressão, ou seja, não estão agrupadas com parênteses.

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

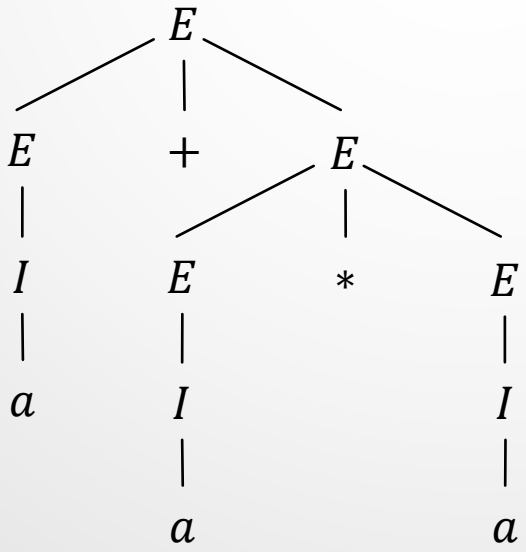
- **Ambiguidade em gramáticas e linguagens:**
  - **Exemplo:** dada a forma sentencial $E + E * E$, existem duas derivações a partir de E:
    1. $E \Rightarrow E + E \Rightarrow E + E * E$
    2. $E \Rightarrow E * E \Rightarrow E + E * E$

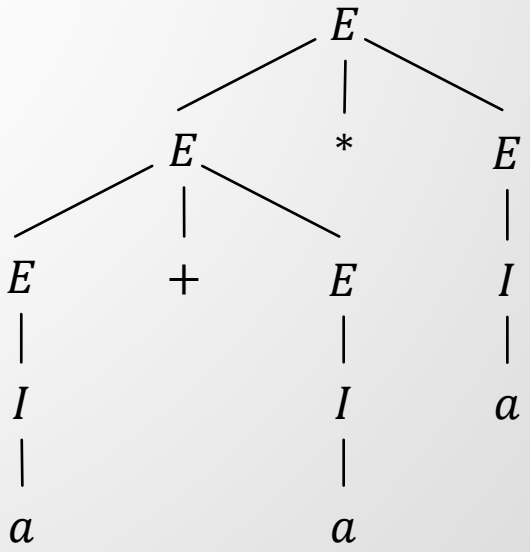




Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Ambiguidade em gramáticas e linguagens:**
  - **Formalmente:** uma CFG $G = (V, T, P, S)$ é ambígua se existe pelo menos uma string w em T^* para a qual podemos encontrar duas árvores de análise sintática diferentes, cada qual com uma raiz identificada por S, e um resultado w. Se cada string tiver no máximo uma árvore de análise sintática, a gramática é não-ambígua.
  - De modo a mostrar que o exemplo anterior realmente identifica a gramática em questão como ambígua, iremos completar as árvores de análise sintática considerando a string $a + a * a$:





Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Ambiguidade em gramáticas e linguagens:**
  - **Remoção de ambiguidade:** Surpreendentemente, não existe absolutamente nenhum algoritmo que possa nos informar se uma CFG é ambígua e também existem, como já mencionado, CFLs que só possuem CFGs ambíguas, tornando impossível a remoção de ambiguidade para essas linguagens;
  - Na prática, a situação não é tão problemática, pois existem técnicas conhecidas para a remoção de alguns tipos de ambiguidade;
  - Voltando novamente ao nosso exemplo da linguagem de expressões, existem duas causas para a ambiguidade da gramática da linguagem:
    1. A precedência dos operadores não é respeitada, sendo assim precisamos impor que a gramática gere apenas uma estrutura, respeitando a precedência;
    2. Uma sequência de operadores idênticos pode ser agrupada a partir da esquerda ou a partir da direita, por exemplo, a expressão $a + a + a$ possui mais de uma árvore de análise sintática. Mesmo que na aritmética possamos calcular qualquer uma das duas adições em primeiro lugar e depois a restante (associatividade), precisamos remover a ambiguidade. Convencionalmente insiste-se no agrupamento à esquerda.

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Ambiguidade em gramáticas e linguagens:**
  - Para o problema de impor precedência, a solução é introduzir muitas variáveis diferentes, de modo a estratificar a estrutura da árvore de análise sintática gerada, "jogando" para baixo ou forçando que expressões com maior precedência apareçam em níveis maiores da árvore (níveis mais abaixo). Para o exemplo das expressões, podemos introduzir os conceitos de **fator** e **termo**:
  - **Fator:** um fator é uma expressão que não pode ser separada por qualquer operador adjacente, seja um $*$ ou um $+$. No nosso exemplo, os únicos fatores são:
    a) identificadores: não é possível separar os símbolos de um identificador associando um operador;
    b) expressões entre parênteses: não importa o que aparece dentro dos parênteses, pois o propósito deles é impedir o que está entre eles de se tornar o operando de qualquer operador fora deles.
  - **Termo:** um termo é uma expressão que não pode ser quebrada pelo operador $+$. No nosso exemplo, um termo é um produto de um ou mais fatores.

Linguagens Livres de Contexto
Gramáticas Livres de Contexto

- **Ambiguidade em gramáticas e linguagens:**
  - Ainda, uma expressão será qualquer expressão possível, inclusive aquelas que podem ser quebradas por um $*$ adjacente ou por um + adjacente. Desse modo, uma expressão para o nosso exemplo é uma soma de um ou mais termos.
  - Como resultado, temos a gramática:

$$
\begin{aligned}
E &\rightarrow T \mid E + T \\
T &\rightarrow F \mid T * F \\
F &\rightarrow I \mid (E) \\
I &\rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1
\end{aligned}
$$

Linguagens Livres de Contexto
Exercícios Escritos

Exercício e7.1: Projete uma gramática livre de contexto para as seguintes linguagens:

a) $L = \{a^n b^n \mid n \geq 1\}$

b) $L = \{\, w \in \{0,1\}^* \mid w \text{ começa e termina com o mesmo símbolo} \,\}$

c) $L = \{a^i b^j c^k \mid i \neq j \text{ ou } j \neq k\}$

d) $L = \{\, w \mid w \text{ é uma expressão balanceada composta por parênteses} \,\}$

e) $L = \{\, w \mid w \text{ é uma expressão balanceada composta por parênteses e colchetes} \,\}$

Linguagens Livres de Contexto
Exercícios Escritos

Exercício e7.2: A gramática a seguir gera a linguagem de expressões regulares $0^*1(0+1)^*$:

$$
\begin{aligned}
S &\rightarrow A1B \\
A &\rightarrow 0A \mid \varepsilon \\
B &\rightarrow 0B \mid 1B \mid \varepsilon
\end{aligned}
$$

Forneça as derivações mais à esquerda, mais à direita e a árvore de análise sintática das seguintes strings:

a) 00101

b) 1001

c) 00011

Linguagens Livres de Contexto
Exercícios Escritos

Exercício e7.3: A gramática a seguir gera expressões prefixadas com os operandos x e y e operadores binários $+$, $-$ e $*$:

$$E \quad \rightarrow \quad +EE \mid -EE \mid *EE \mid x \mid y$$

Forneça as derivações mais à esquerda, mais à direita e a árvore de análise sintática da string $+* -xyxy$

Linguagens Livres de Contexto
Exercícios Escritos

Exercício e7.4: A gramática a seguir é ambígua:

$$S \quad \rightarrow \quad aS \mid aSbS \mid \varepsilon$$

Mostre que, em particular, a string aab tem duas:

a) Árvores de análise sintática;

b) Derivações mais à esquerda;

c) Derivações mais à direita.

Exercício e7.5: Encontre uma gramática não-ambígua para a gramática do exercício anterior.

Linguagens Livres de Contexto
Exercícios Escritos

Exercício e7.6: Dada a gramática não-ambígua:

$$
\begin{aligned}
E &\rightarrow T \mid E + T \\
T &\rightarrow F \mid T * F \\
F &\rightarrow I \mid (E) \\
I &\rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1
\end{aligned}
$$

Estenda-a, mantendo-a não-ambígua, para gerar expressões que também contenham os operadores de subtração e divisão, representados, respectivamente pelos símbolos – e /. Forneça a árvore de análise sintática para a string $a + (b * a)/a0 - b0$

Bibliografia

HOPCROFT, J. E.; ULLMAN, J. D.; MOTWANI, R. **Introdução à Teoria de Autômatos, Linguagens e Computação**. 2. ed. Rio de Janeiro: Elsevier, 2002. 560 p.

RAMOS, M. V. M.; JOSÉ NETO, J.; VEGA, I. S. **Linguagens Formais: Teoria, Modelagem e Implementação**. Porto Alegre: Bookman, 2009. 656 p.

SIPSER, M. **Introdução à Teoria da Computação**. 2. ed. São Paulo: Cengage Learning, 2017. 459 p.