Gramáticas

O texto apresentado neste documento é uma adaptação de:

Newton Vieira, 2006. Introdução aos fundamentos da computação: Linguagens e máquinas. CENGAGE Learning.

No texto em que vamos nos basear para este tópico, usa-se o termo “palavra” em vez de “string”. Ou seja, poderemos utilizar os termos *string* ou *palavra* para nos referir a uma sequência de comprimento finito de símbolos de um determinado alfabeto.

Até o momento, estudamos duas ferramentas formais para especificação de linguagens:

- Definições recursivas de conjuntos.
- Definições por meio de operações sobre conjuntos (conjuntos/expressões regulares).

Agora estudaremos uma terceira abordagem, conhecida como *Gramática*, que usa como princípio para definição de linguagens a substituição de símbolos em palavras, para formar outras palavras.

1. Introdução

Antes de dar uma definição precisa de gramática, serão apresentados os conceitos envolvidos de maneira informal. Como vimos anteriormente, uma definição recursiva provê um meio de construir (ou gerar) os elementos de um conjunto (enumerável). Similarmente, uma gramática mostra como gerar as palavras de uma linguagem.

O elemento fundamental das gramáticas é a regra (também chamada de regra de produção). Uma regra é um par ordenado (u, v), tradicionalmente escrito na forma u → v, onde u e v são palavras que podem conter símbolos de dois alfabetos disjuntos, um com símbolos denominados **variáveis**, ou **não terminais**, e outro com símbolos denominados **terminais**. As variáveis são símbolos auxiliares para a geração das palavras da linguagem, enquanto que os terminais são o alfabeto da linguagem definida. Nos exemplos a seguir, usamos letras maiúsculas para variáveis e minúsculas para terminais. Segue um exemplo de regra:

aAB → baA

Tal regra especifica que: dada uma palavra que contenha a subpalavra aAB, tal subpalavra pode ser substituída por baA. Assim, a partir da palavra aABBaAB, aplicando-se a regra acima pode-se obter (diz-se **derivar**):

- baABaAB, substituindo a primeira ocorrência de aAB;
- aABBbaA, substituindo a segunda ocorrência de aAB.

A relação de derivação é denotada por ⇒. Por exemplo, a cadeia de derivações a seguir mostra uma derivação de bbaAbaA a partir de aABBaAB:

aABBaAB ⇒ baABaAB (aplicando-se a regra aAB → baA)
⇒ bbaAaAB (aplicando-se a regra aAB → baA)
⇒ bbaAbaA (aplicando-se a regra aAB → baA)

Uma gramática é composta por um conjunto de regras e da indicação de uma variável especial denominada variável de partida. Qualquer derivação deve começar com a palavra constituída apenas da variável de partida.

As palavras de variáveis e/ou terminais geradas a partir da variável de partida são chamadas **formas sentenciais** da gramática. Assim, uma regra pode ser aplicada a uma forma sentencial para gerar uma outra forma sentencial. Uma forma sentencial sem variáveis é denominada **sentença**. A linguagem definida pela gramática, também dita gerada pela gramática, é o conjunto de sentenças geradas pela gramática. Para uma gramática G, a linguagem gerada por ela é denotada por L(G).

Exemplo 1.1

Seja a gramática G constituída da variável de partida P e das regras:

1. $P \rightarrow \mathtt{a}A\mathtt{bc}$
2. $A \rightarrow \mathtt{a}A\mathtt{b}C$
3. $A \rightarrow \lambda$
4. $C\mathtt{b} \rightarrow \mathtt{b}C$
5. $C\mathtt{c} \rightarrow \mathtt{cc}$

Toda derivação de G deve começar com a forma sentencial constituída da variável de partida P. Um exemplo de derivação:

P ⇒ aAbc (regra 1)
⇒ abc (regra 3)

Isto mostra que abc é uma sentença da linguagem gerada por G: abc ∈ L(G). Um outro exemplo de derivação:

P ⇒ aAbc (regra 1)
⇒ aaAbCbc (regra 2)
⇒ aaaAbCbCbc (regra 2)
⇒ aaabCbCbc (regra 3)
⇒ aaabbCCbc (regra 4)
⇒ aaabbCbCc (regra 4)
⇒ aaabbCbcc (regra 5)
⇒ aaabbbCcc (regra 4)
⇒ aaabbbccc (regra 5)

Logo, aaabbbccc ∈L(G). Na verdade, pode-se mostrar que

$$L(G) = \{\mathtt{a}^n\mathtt{b}^n\mathtt{c}^n \mid n \geq 1\}.$$

❑

Agora, define-se formalmente o que é gramática.

Definição 1.1

Uma gramática é uma quádrupla (V, Σ, R, P), onde:

- V é um conjunto finito de elementos denominados variáveis;
- Σ é um alfabeto; V ∩ Σ = ∅;
- $R \subseteq (V \cup \Sigma)^+ \times (V \cup \Sigma)*$ é um conjunto finito de pares denominados regras; e
- P ∈ V é uma variável denominada variável de partida.

Observe que o lado esquerdo de uma regra não pode ser λ.

Seja uma gramática G = (V, Σ, R, P). Diz-se que x ⇒ y em G, onde x, y ∈ (V U Σ)*, se há uma regra u → v ∈ R tal que u ocorre em x e y é o resultado de substituir uma ocorrência de u em x por v. A relação $\overset{n}{\Rightarrow}$ é definida recursivamente assim, utilizando-se ⇒:

- $x \overset{0}{\Rightarrow} x$ para todo x ∈ (V U Σ)*;
- se $w \overset{n}{\Rightarrow} xuy$ e u → v ∈ R então $w \overset{n+1}{\Longrightarrow} xvy$ para todo w, x, y ∈ (V U Σ)*, n ≤ 0.

Quando $x \overset{n}{\Rightarrow} y$, diz-se que "de x deriva-se y em n passos", ou então que "há uma derivação de tamanho n de y a partir de x". Diz-se ainda que de x deriva-se y em zero ou mais passos, $x \overset{*}{\Rightarrow} y$, se existe n ≥ 0 tal que $x \overset{n}{\Rightarrow} y$. E de x deriva-se y em um ou mais passos, $x \overset{+}{\Rightarrow} y$, se existe n ≥ 1 tal que $x \overset{n}{\Rightarrow} y$. Com isto, pode-se definir o que é a linguagem gerada pela gramática G:

$$L(G) = \{w \in \Sigma^* \mid P \overset{*}{\Rightarrow} w\}.$$

Exemplo 1.2

Seja a gramática G do Exemplo 1.1.

As duas derivações demonstram que $P \overset{2}{\Rightarrow} abc$ e $P \overset{9}{\Rightarrow} aaabbbccc$.

É fácil mostrar que todas as palavras da forma $a^n b^n c^n$, para n ≥ 1, são geradas por G. Basta notar que tais palavras podem ser geradas por derivações da forma:

$$\begin{aligned}
P &\Rightarrow aAbc && \text{(regra 1)}\\
&\overset{k}{\Rightarrow} aa^kA(bC)^kbc && \text{(regra 2, } k \text{ vezes; } k \geq 0)\\
&\Rightarrow aa^k(bC)^kbc && \text{(regra 3)}\\
&\Rightarrow a^{k+1}(bC)^{k-1}b^2Cc && \text{(regra 4, 1 vez)}\\
&\overset{2}{\Rightarrow} a^{k+1}(bC)^{k-2}b^3C^2c && \text{(regra 4, 2 vezes)}\\
&\vdots\\
&\overset{k}{\Rightarrow} a^{k+1}b^{k+1}C^kc && \text{(regra 4, } k \text{ vezes)}\\
&\overset{k}{\Rightarrow} a^{k+1}b^{k+1}c^{k+1} && \text{(regra 5, } k \text{ vezes)}
\end{aligned}$$

Logo, conclui-se que

$$\{a^nb^nc^n \mid n \geq 1\} \subseteq L(G).$$

❑

No Exemplo 1.2, mostrou-se como provar que um conjunto A está contido na linguagem gerada por uma gramática G, ou seja, A ⊆ L(G), mediante o que se denomina um esquema de derivação. Tal esquema indica como as palavras de A podem ser geradas pela gramática G. Veremos mais tarde como usar indução matemática para executar tarefas como essa.

A mesma linguagem pode ser gerada por inúmeras gramáticas. Duas gramáticas G e G' são ditas equivalentes quando L(G) = L(G'). O problema de modificar uma gramática de forma que ela atenda a certos requisitos, mas sem alterar a linguagem gerada pela mesma, é importante em certos contextos, como, por exemplo, na construção de analisadores sintáticos de linguagens. Algumas técnicas de manipulação de gramáticas serão abordadas em aulas seguintes.

É muito comum uma gramática ter duas ou mais regras com o mesmo lado esquerdo. Neste caso, pode ser útil abreviar colocando-se apenas um lado esquerdo e colocando-se os lados direitos das várias regras separadas por "|". Exemplo:

$$A \rightarrow \mathrm{a}A\mathrm{b}C \mid \lambda$$

No exemplo a seguir, as regras têm a característica que os seus lados esquerdos contêm apenas e tão somente uma variável. Este tipo de gramática é muito importante em termos práticos, e será estudado em detalhes mais adiante.

Exemplo 1.3

Seja a gramática G = (V, Σ,R,E), onde:

- $V = \{E, T, N, D\}$
- $\Sigma = \{+, -, (,), 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
- R contém as regras:

$$E \rightarrow E + T \mid E - T \mid T$$
$$T \rightarrow (E) \mid N$$
$$N \rightarrow DN \mid D$$
$$D \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

A gramática G gera expressões aritméticas contendo números na base decimal, operadores de soma e subtração, e parênteses. Através das três regras com E do lado esquerdo, pode-se gerar uma sequência de somas e/ou subtrações de T's (termos); exemplo:

$$\begin{aligned} E &\Rightarrow E + T && (\text{regra } E \rightarrow E + T) \\ &\Rightarrow E - T + T && (\text{regra } E \rightarrow E - T) \\ &\Rightarrow E - T - T + T && (\text{regra } E \rightarrow E - T) \\ &\Rightarrow T - T - T + T && (\text{regra } E \rightarrow T) \end{aligned}$$

Observe como as regras são recursivas à esquerda, levando à produção de uma sequência da direita para a esquerda. Por outro lado, as regras responsáveis pela produção dos números das

expressões são recursivas à direita, redundando na produção de números da esquerda para a direita. Por exemplo, para gerar um número de quatro dígitos pode-se derivar:

$$\begin{aligned} N &\Rightarrow DN && (\text{regra } N \to DN) \\ &\Rightarrow DDN && (\text{regra } N \to DN) \\ &\Rightarrow DDDN && (\text{regra } N \to DN) \\ &\Rightarrow DDDD && (\text{regra } N \to D) \end{aligned}$$

e, em seguida, derivar os quatro dígitos usando-se as regras com D do lado esquerdo.

Observe também que a geração de parênteses se dá através de recursão (no caso, indireta), a qual não pode ser classificada como recursão à esquerda nem à direita. Por exemplo, na derivação:

$$\begin{aligned} E &\Rightarrow E + T && (\text{regra } E \to E + T) \\ &\Rightarrow T + T && (\text{regra } E \to T) \\ &\Rightarrow (E) + T && (\text{regra } T \to (E)) \end{aligned}$$

a variável E aparece (recursivamente) na forma sentencial entre "(" e ")". ❑

2. Gramáticas Livres de Contexto

Conforme antecipado na Seção 1, existe um tipo de gramática muito importante em termos práticos, que são aquelas cujo lado esquerdo das regras de produção é formado sempre exatamente por um único símbolo não terminal. São chamadas *Gramáticas Livres de Contexto*.

Definição 2.1

Uma *Gramática Livre de Contexto* (GLC) é uma gramática (V, Σ, R, P), em que cada regra tem a forma $X \to w$, onde $X \in V$ e $w \in (V \cup \Sigma)^*$.

Para uma GLC, em cada passo de uma derivação deve-se escolher, na forma sentencial, a variável A a ser substituída pelo lado direito de uma regra que tenha A como lado esquerdo. Ao se fazer tal substituição, diz-se que A é *expandida*.

Exemplo 2.1

A linguagem $\{0^n 1^n \mid n \in N\}$ é gerada pela GLC G = ({P}, {0, 1}, R, P), onde R é o conjunto formado pelas duas regras:

$$P \to 0P1 \mid \lambda$$

As únicas palavras geradas por esta gramática são aquelas que podem ser geradas por n aplicações da regra P → 0P1, n ≥ 0, seguidas de uma aplicação da regra P → λ.
Esquematicamente:

$$P \overset{n}{\Rightarrow} 0^n P 1^n \Rightarrow 0^n 1^n$$

logo L(G) = $\{0^n 1^n \mid n \in N\}$. ❑

Exemplo 2.2

Seja G = ({P}, {0, 1}, R, P) onde R é formado pelas 5 regras a seguir:

$$P \to 0P0 \mid 1P1 \mid 0 \mid 1 \mid \lambda$$

Esta gramática gera os palíndromos sobre {0, 1}.

Aplicando-se as duas primeiras regras, gera-se qualquer forma sentencial do tipo wPw^R, onde w^R é o reverso da palavra w, para w ∈ {0, 1}. Por fim, para gerar uma palavra, aplica-se uma das 3 últimas regras; a última, quando a palavra tem comprimento par, e uma das outras duas, quando ela tem comprimento ímpar. ❑

Exemplo 2.3

A linguagem L = {w ∈ {0,1}* | w tem um número igual de 0's e 1's} pode ser gerada pela gramática G = ({P}, {0, 1}, R, P), onde R é formado pelas 3 regras a seguir:

$$P \to 0P1P \mid 1P0P \mid \lambda$$

Como o lado direito de cada uma das 3 regras tem número igual de 0's e 1's, G só gera palavras de L. Para mostrar que G gera **todas** as palavras de L, podem ser analisados 3 casos:

(a) w = λ; neste caso, basta aplicar a regra P → λ;
(b) w = 0y para algum y ∈ {0, 1}* e y tem um 1 a mais que 0's; neste caso, y pode ser escrito na forma x1z, onde x tem número igual de 0's e 1's e z também tem número igual de 0's e 1's; assim, basta iniciar a derivação de w com a primeira regra;
(c) w = 1y para algum y ∈ {0, 1}* e y tem um 0 a mais que 1's; por motivo análogo ao segundo caso, basta aplicar a segunda regra.

Nos casos (b) e (c), tem-se novamente (recursivamente) os três casos aplicados para as subpalavras x e z. Com isto, tem-se um método para construir uma derivação de qualquer palavra de L. A seguinte derivação de 01010110 ilustra a aplicação do método subjacente, onde a variável expandida é sempre a mais à esquerda:

$$\begin{aligned}
P &\Rightarrow 0P1P && P \to 0P1P\ (x = 10;\ y = 0110)\\
&\Rightarrow 01P0P1P && P \to 1P0P\ (x = \lambda;\ y = \lambda)\\
&\Rightarrow 010P1P && P \to \lambda\\
&\Rightarrow 0101P && P \to \lambda\\
&\Rightarrow 01010P1P && P \to 0P1P\ (x = \lambda;\ y = 10)\\
&\Rightarrow 010101P && P \to \lambda\\
&\Rightarrow 0101011P0P && P \to 1P0P\ (x = \lambda;\ y = \lambda)\\
&\Rightarrow 01010110P && P \to \lambda\\
&\Rightarrow 01010110 && P \to \lambda.
\end{aligned}$$

❑

O exemplo a seguir ilustra uma gramática que contém a essência da especificação da sintaxe das expressões aritméticas das linguagens de programação usuais.

Exemplo 2.4

Seja a GLC ({E, T, F}, {t,+, _, (,)}, R, E), onde R é formado pelas regras:

$$E \rightarrow E{+}T \mid T$$

$$T \rightarrow T{*}F \mid F$$

$$F \rightarrow (E) \mid \mathtt{t}$$

As duas primeiras regras dizem que uma expressão aritmética E é constituída de um ou mais termos T's somados. As duas seguintes dizem que um termo é constituído de um ou mais fatores F's multiplicados. E as duas últimas dizem que um fator é terminal t ou, recursivamente, uma expressão aritmética entre parênteses. ❑

Uma simplificação para a especificação de gramáticas livres de contexto é frequentemente usada, quando se apresenta apenas as regras de produção, sem deixar explícito quais são os símbolos terminais, não terminais e o símbolo de partida. Em casos assim, vamos assumir que as letras maiúsculas vão denotar símbolos não terminais, letras minúsculas e demais símbolos vão denotar símbolos terminais, e o símbolo que aparecer do lado esquerdo na primeira regra em uma GLC será o símbolo de partida. Assim, a gramática do Exemplo 2.4 pode ser simplesmente descrita pelas suas regras de produção:

$$E \rightarrow E{+}T \mid T$$

$$T \rightarrow T{*}F \mid F$$

$$F \rightarrow (E) \mid \mathtt{t}$$

Essa simplificação pode ser aplicada em outras gramáticas que vimos neste material. Por exemplo, a gramática do Exemplo 2.1 poderia simplesmente ser especificada por:

$$P \rightarrow \mathtt{0}P\mathtt{1} \mid \lambda$$

Definição 2.2

Uma linguagem é dita ser uma *linguagem livre do contexto* se existe uma gramática livre do contexto que a gera.