Capítulo 5

Gramáticas Geradoras

As gramáticas geradoras surgiram na década de 50 do século XX, com o importante estudo do linguista Noam Chomsky (Estados Unidos da América, 1928 -) sobre a formalização das linguagens naturais. As gramáticas permitem capturar parte da sintaxe das linguagens naturais sendo o instrumento ideal para lidar com a sintaxe das linguagens de programação.

Vamos estudar a noção de gramática geradora de uma linguagem, centrando-nos em especial na classe das gramáticas independentes do contexto.

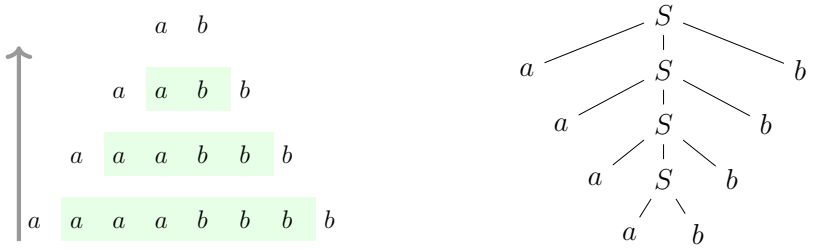
5.1 Gramáticas geradoras

Vimos que os autómatos são mecanismos que permitem reconhecer linguagens. Por outro lado, as expressões regulares permitem especificar formalmente linguagens, tendo também subjacente um mecanismo de reconhecimento que advém da especificação de um algoritmo de concordância de palavras com os padrões das expressões regulares (essencialmente um autómato finito).

Uma gramática geradora é, em certo sentido, uma generalização de uma expressão regular. Uma gramática especifica um conjunto finito de regras que permitem gerar as palavras de uma linguagem, usando substituições sucessivas de padrões.

Começamos com um exemplo ilustrativo de uma gramática para gerar a linguagem não regular $L = \{a^n b^n : \ n \in \mathbb{N}\}$. Precisamos de definir um conjunto finito de regras, designadas produções, envolvendo os símbolos do alfabeto $\{a, b\}$ e variáveis auxiliares, as quais permitem usar um simples método de substituição para gerar qualquer palavra da linguagem.

Na figura seguinte, à esquerda, ilustramos como derivar a palavra a^4b^4 usando uma recorrência da forma $S = aSb$, onde S é uma variável, e parando na palavra base $S = ab$. No lado direito, podemos constatar que a derivação corresponde à construção de uma árvore de raiz S.



Assim, para derivar qualquer palavra desta linguagem basta considerar duas regras de derivação (produções): a primeira, $S \to ab$, permite derivar a palavra ab partindo de S; a segunda, $S \to aSb$, permite derivar sucessivamente sentenças da forma $a^k S b^k$. Quando combinadas sequencialmente, estas duas produções permitem gerar todas palavras da forma $a^n b^n$.

Definição 5.1.1 Gramática Geradora.

Uma **gramática geradora** é um quádruplo ordenado $G = (V, T, P, S)$, onde:

- V e T são conjunto finitos não vazios e disjuntos de símbolos;
- V é o conjunto das *variáveis*;
- T é o conjunto dos símbolos *terminais*;
- P é um conjunto finito de *produções* (regras de reescrita) da forma

  $$\alpha A \beta \to \gamma$$

  em que $A \in V$ e $\alpha, \beta, \gamma \in (V \cup T)^\star$;
- $S \in V$ é o *axioma da gramática* (variável inicial).

O **vocabulário** de uma gramática $G = (V, T, P, S)$ é o conjunto das variáveis e dos símbolos terminais, $V \cup T$. Vamos assumir que o símbolo de produção "$\to$" não pertence ao vocabulário. Uma **sentença** é uma sequência de símbolos do vocabulário, isto é, um elemento de $(V \cup T)^\star$. Uma **palavra** é uma sequência de símbolos terminais, ou seja, um elemento de $T^\star$.

Formalizamos agora a noção de derivação no contexto de uma gramática geradora, a qual permite construir palavras a partir do axioma.

Definição 5.1.2 Derivação.

Consideremos uma gramática $G = (V, T, P, S)$ e sejam $\alpha, \beta, \gamma, \mu \in (V \cup T)^\star$.

Derivação num só passo: A sentença $\alpha\mu\beta$ é derivável a partir da sentença $\alpha\gamma\beta$ num só passo, e escrevemos

$$\alpha\gamma\beta \Rightarrow \alpha\mu\beta$$

sempre que $\gamma \to \mu$ seja uma produção em P.

Derivação em zero ou mais passos: A sentença β é derivável a partir da sentença α, e escrevemos

$$\alpha \overset{\star}{\Rightarrow} \beta$$

sempre que $\alpha = \beta$ ou exista γ tal que $\alpha \overset{\star}{\Rightarrow} \gamma$ e $\gamma \Rightarrow \beta$.

A relação $\overset{\star}{\Rightarrow}$ é o fecho reflexivo e transitivo de $\Rightarrow$. Usamos o símbolo $\overset{+}{\Rightarrow}$ para denotar uma derivação em um ou mais passos. A relação $\overset{+}{\Rightarrow}$ é apenas o fecho transitivo de $\Rightarrow$.

As palavras deriváveis a partir do axioma de uma gramática constituem naturalmente a linguagem gerada por essa gramática.

Definição 5.1.3 Linguagem Gerada.

A *linguagem gerada* por uma gramática $G = (V, T, P, S)$ é

$$\mathcal{L}(G) = \{w \in T^\star : \ S \overset{\star}{\Rightarrow} w\}.$$

A única restrição que é imposta quanto à forma das produções numa gramática geradora é que o lado esquerdo de cada produção contenha pelo menos uma variável. Assim, as regras de derivação podem ser mais ou menos complexas, conforme ilustramos no exemplo que se segue.

Exemplo 5.1.4

Consideremos a gramática $G = (\{S, X, Y\}, \{a, b, c\}, P, S)$, com as seguintes produções P:

$$\begin{array}{lll} S \to abc & S \to aXbc & \\ Xb \to bX & Xc \to Ybcc & \\ bY \to Yb & aY \to aaX & aY \to aa. \end{array}$$

A partir de S derivamos num só passo a palavra abc, usando a produção $S \to abc$.

A única alternativa que permite derivar outras palavras é aplicar a produção $S \to aXbc$. Temos assim que $S \Rightarrow aXbc$ e só é possível continuar a derivação se aplicarmos a produção $Xb \to bX$, obtendo $S \overset{\star}{\Rightarrow} abXc$. Prosseguindo, apenas podemos usar a produção $Xc \to Ybcc$, obtendo $S \overset{\star}{\Rightarrow} abYbcc$. Em seguida, só é possível usar a produção $bY \to Yb$, obtendo $S \overset{\star}{\Rightarrow} aYbbcc$.

A partir daqui, podemos seguir uma de duas alternativas. Na primeira, aplicamos a produção $aY \to aa$, obtendo a palavra $aabbcc$. Na segunda, usamos a produção $aY \to aaX$, obtendo a sentença $aaXbbcc$.

Continuando este processo, derivamos sucessivamente palavras $a^n b^n c^n$ e sentenças $a^n X b^n c^n$, para $n = 2, 3, \ldots$

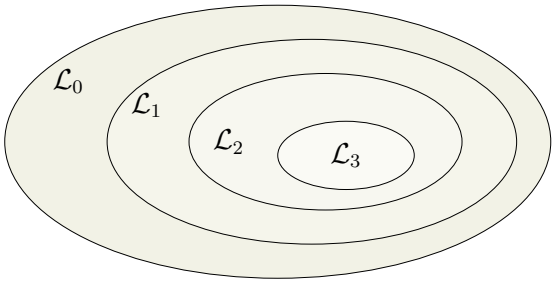
Assim, por indução, mostramos que a linguagem gerada por G é

$$\mathcal{L}(G) = \{a^n b^n c^n : \ n \in \mathbb{N}\}.$$

A imposição de restrições adicionais quanto à forma das produções permite definir várias classes de gramáticas e as correspondentes classes de linguagens. Chomsky considerou quatro classes principais de linguagens, $\mathcal{L}_0$, $\mathcal{L}_1$, $\mathcal{L}_2$ e $\mathcal{L}_3$, associadas a quatro tipos de gramáticas:

- Tipo 0 - Gramáticas com Estrutura de Frase - Não são impostas quaisquer restrições às produções das gramáticas. Esta é a classe mais geral, associada às gramáticas das linguagens naturais.
- Tipo 1 - Gramáticas Sensíveis ao Contexto - Nestas gramáticas, o tamanho do lado esquerdo de cada produção não excede o tamanho do lado direito. Mais precisamente, para qualquer produção $\alpha \to \beta$ diferente de $S \to \varepsilon$, tem-se que $|\alpha| \leq |\beta|$. Para além disso, se $S \to \varepsilon$ é uma produção da gramática então o símbolo S não aparece no lado direito das produções.
- Tipo 2 - Gramáticas Independentes do Contexto - Nestas gramáticas, o lado esquerdo de cada uma das produções é uma variável.
- Tipo 3 - Gramáticas Regulares - São gramáticas que permitem gerar as linguagens regulares.

Conforme ilustrado na figura seguinte, temos que $\mathcal{L}_0 \supset \mathcal{L}_1 \supset \mathcal{L}_2 \supset \mathcal{L}_3$, sendo todas estas inclusões estritas.



Vamos tratar agora do estudo das gramáticas do tipo 2 e na Secção 5.3, p. 141 estudaremos as gramáticas do tipo 3. A análise das gramáticas dos tipos 0 e 1 ultrapassa o âmbito introdutório deste livro. Sugerimos a consulta de [3] ou [2] para mais detalhes sobre os fundamentos e aplicações das gramáticas desses tipos.

Definição 5.1.5 Gramática Independente do Contexto.

Uma gramática geradora é **independente do contexto** (GIC) se as suas produções têm a forma

$$A \to \alpha$$

com $A \in V$ e $\alpha \in (V \cup T)^\star$.

Definição 5.1.6 Linguagem Independente do Contexto.

Uma linguagem é independente do contexto (LIC) quando existe alguma gramática independente do contexto que a gera.

Exemplo 5.1.7

A gramática independente do contexto $G = (\{S\}, \{0,1\}, P, S)$ com as produções

$$\begin{array}{lll} S \to \varepsilon & S \to 0 & S \to 1 \\ S \to 0S0 & S \to 1S1 & \end{array}$$

define a linguagem dos palíndromos binários.

É usual combinar produções que tenham o mesmo lado esquerdo recorrendo ao símbolo "|" com o significado de "ou".

Para este exemplo, a lista de produções pode ser apresentada de forma compacta como

$$S \to \varepsilon \mid 0 \mid 1 \mid 0S0 \mid 1S1$$

Exemplo 5.1.8

Consideremos a gramática de Expressões Aritméticas (E), sobre Números Binários (N) e Identificadores (I), só com as letras a e b, usando apenas os Operadores $+$ e $\times$,

$$G = (\{E, I, N\}, \{a, b, 0, 1, (,), +, \times, -\}, P, E)$$

onde P é o conjunto de produções

$$\begin{aligned} E &\to I \mid N \mid E + E \mid E \times E \mid (E) \\ I &\to a \mid b \mid Ia \mid Ib \\ N &\to 0 \mid 1 \mid N0 \mid N1 \mid -N \mid +N \end{aligned}$$

Será que a palavra $(a + ab) \times -100$ é uma Expressão Aritmética?

Vejamos se é possível atingir uma sequência de símbolos terminais igual aquela palavra, partindo do símbolo inicial e por derivações passo a passo:

$$\begin{aligned} E &\Rightarrow E \times E \Rightarrow E \times N \Rightarrow E \times -N \Rightarrow E \times -N0 \Rightarrow E \times -N00 \\ &\Rightarrow E \times -100 \Rightarrow (E) \times -100 \Rightarrow (E + E) \times -100 \\ &\Rightarrow (E + I) \times -100 \Rightarrow (E + Ib) \times -100 \\ &\Rightarrow (E + ab) \times -100 \Rightarrow (I + ab) \times -100 \Rightarrow (a + ab) \times -100 \end{aligned}$$

Logo, $(a + ab) \times -100$ é uma Expressão Aritmética.

As gramáticas independentes do contexto são utilizadas para especificar a sintaxe da maioria das linguagens de programação. Com base na gramática de uma linguagem de programação são construídos analisadores sintácticos ("parsers") os quais permitem detectar e identificar erros de sintaxe nos programas escritos nessa linguagem.

Exemplo 5.1.9

Produções numa GIC para as instruções compostas na linguagem C++

```
<compound stmt> ::= { <stmt list> }
<stmt list> ::= <stmt> <stmt list> | epsilon
<stmt> ::= <compound stmt>
<stmt> ::= if ( <expr> ) <stmt>
<stmt> ::= if ( <expr> ) <stmt> else <stmt>
<stmt> ::= while ( <expr> ) <stmt>
<stmt> ::= do <stmt> while ( <expr> ) ;
<stmt> ::= for ( <stmt> <expr> ; <expr> ) <stmt>
<stmt> ::= case <expr> : <stmt>
<stmt> ::= switch ( <expr> ) <stmt>
<stmt> ::= break ; | continue ;
<stmt> ::= return <expr> ; | goto <id> ;
```

Em geral, provar analiticamente que uma gramática gera exactamente uma determinada linguagem é um problema de difícil resolução. A forma especial das produções nas gramáticas independentes do contexto permite que o problema seja tratável, conforme ilustramos no exemplo seguinte.

Exemplo 5.1.10

Consideremos a gramática $G = (\{S\}, \{0,1\}, P, S)$ com as produções

$$S \to \varepsilon \mid 0 \mid 1 \mid 0S0 \mid 1S1$$

e seja $\text{Pal} = \{w \in \{0,1\}^\star : \ w = w^{-1}\}$. Vamos provar que $\text{Pal} = \mathcal{L}(G)$.

$\text{Pal} \subseteq \mathcal{L}(G)$. Para qualquer palíndromo w provamos, por indução sobre $|w|$, que $w \in \mathcal{L}(G)$, ou seja, que $S \overset{\star}{\Rightarrow} w$.

Caso base. Se $|w| = 0$ ou $|w| = 1$ então $w = \varepsilon$, $w = 0$ ou $w = 1$ e $S \Rightarrow \varepsilon$, $S \Rightarrow 0$ e $S \Rightarrow 1$.

Passo indutivo. Como hipótese de indução, admitamos que quando $x = x^{-1}$ então também $S \overset{\star}{\Rightarrow} x$, isto para qualquer palavra x de tamanho $|x| \leq n$.

Seja w uma qualquer palavra de tamanho $|w| = n + 1 \geq 2$, tal que $w = w^{-1}$. Então $w = 0x0$ ou $w = 1x1$, com $x = x^{-1}$ e $|x| = n - 1$. Pela hipótese de indução $S \overset{\star}{\Rightarrow} x$, portanto $S \Rightarrow 0S0 \overset{\star}{\Rightarrow} 0x0$ ou $S \Rightarrow 1S1 \overset{\star}{\Rightarrow} 1x1$. Em qualquer dos casos, concluímos que $S \overset{\star}{\Rightarrow} w$.

$\text{Pal} \supseteq \mathcal{L}(G)$. Seja $w \in \mathcal{L}(G)$, isto é, $S \overset{\star}{\Rightarrow} w$. Provamos por indução sobre o número de passos de derivação que w é um palíndromo.

Caso base. As derivações num só passo são $S \Rightarrow \varepsilon$, $S \Rightarrow 0$ ou $S \Rightarrow 1$. Em qualquer dos casos $w \in \text{Pal}$.

Passo indutivo. Como hipótese de indução, admitamos que quando $S \overset{\star}{\Rightarrow} x$ em n passos então $x = x^{-1}$, isto para qualquer palavra x.

Uma derivação em $n+1$ passos da palavra w só pode ser da forma $S \Rightarrow 0S0 \overset{\star}{\Rightarrow} 0x0 = w$ ou $S \Rightarrow 1S1 \overset{\star}{\Rightarrow} 1x1 = w$, onde a derivação da palavra x é em n passos. Em qualquer dos casos, pela hipótese de indução $x = x^{-1}$ e portanto $w = w^{-1}$.

5.2 Simplificação e normalização de GIC

Nesta secção vamos obter uma forma normal para as gramáticas independentes do contexto, simplificando e restringindo os lados direitos das produções.

5.2.1 Eliminação de Símbolos Inúteis

As produções de uma gramática podem incluir variáveis que não servem para derivar qualquer palavra, bem como símbolos terminais que não fazem parte das palavras derivadas, sendo portanto inúteis. Podemos eliminar da gramática todas as produções que contenham símbolos ou variáveis inúteis, obtendo uma gramática mais simples e ainda equivalente à gramática original.

Definição 5.2.1 Símbolo Útil.

Um símbolo $X \in V \cup T$ é **útil** se existir uma derivação da forma

$$S \overset{\star}{\Rightarrow} \alpha X \beta \overset{\star}{\Rightarrow} w$$

com $w \in T^\star$ e $\alpha, \beta \in (V \cup T)^\star$.

Portanto, um símbolo X é útil se

1. for **atingível**: existem $\alpha, \beta \in (V \cup T)^\star$ tais que $S \overset{\star}{\Rightarrow} \alpha X \beta$;
2. e se for **gerador**: existem $X \in V$ e $w \in T^\star$ tal que $X \overset{\star}{\Rightarrow} w$.

Um símbolo **inútil** é um símbolo que não é útil e uma produção é inútil se contém um símbolo inútil.

Resulta da definição anterior que as únicas produções que importa reter numa gramática são aquelas em que todos os símbolos são simultâneamente geradores e atingíveis.

Algoritmo 5.2.2 Eliminação de Símbolos Inúteis.

Entrada: uma GIC.

Saída: uma GIC sem símbolos ou produções inúteis.

1. Eliminar todas as produções que contenham símbolos não geradores;
2. Eliminar todas as produções que contenham símbolos não atingíveis.

O algoritmo seguinte determina as variáveis geradoras de uma gramática, aplicando o processo indutivo definido por: se $A \to \alpha$ é uma produção e todos os símbolos em α são geradores então A é uma variável geradora.

Algoritmo 5.2.3 Determinação do Conjunto de Variáveis Geradoras.

Entrada: uma GIC.

Saída: conjunto de variáveis geradoras.

Marcar todos os símbolos terminais como geradores;

Marcar o símbolo ε como gerador;

Repetir

Para cada variável A não marcada

Para cada produção $A \to \alpha$ enquanto A não marcada

Se todos os símbolos em α são geradores

então marcar A como geradora;

até que nenhuma nova variável seja marcada.

Exemplo 5.2.4

Para a gramática G com produções

$$S \to AB \mid C \qquad A \to 0B \mid C$$
$$B \to 1 \mid A0 \qquad C \to AC \mid C1$$

verificamos que:

- Os símbolos terminais 0 e 1 são geradores;
- Por $B \to 1$, a variável B é geradora;
- Por $A \to 0B$, a variável A é geradora;
- Por $S \to AB$, a variável S é geradora.

Assim, o símbolo C é inútil e pode ser eliminado. As novas produções são:

$$S \to AB \qquad A \to 0B \qquad B \to 1 \mid A0$$

O algoritmo seguinte determina o conjunto dos símbolos atingíveis de uma gramática, aplicando o processo indutivo definido por: se $A \to \alpha$ e A é atingível então todos os símbolos em α são atingíveis.

Algoritmo 5.2.5 Determinação do Conjunto de Símbolos Atingíveis.

Entrada: uma GIC $G = (V, T, P, S)$.

Saída: conjunto de símbolos atingíveis.

Marcar a variável S como atingível;

Repetir

  Para cada produção $A \to \alpha$ com A marcada

    Marcar todos os símbolos não marcados em α como atingíveis;

até que nenhum novo símbolo seja marcado.

Exemplo 5.2.6

Para a gramática G com produções

$$S \to AB \qquad A \to 0B$$
$$B \to 1B \mid 0 \qquad C \to AC \mid B1$$

a variável C não é atingível e pode ser eliminada, bem como as produções $C \to AC \mid B1$.

Se o conjunto das variáveis geradoras é vazio então a linguagem gerada pela gramática é também a linguagem vazia. Assim, o problema de decidir se a linguagem gerada por uma GIC é ou não vazia é resolúvel usando o Algoritmo 5.2.3, p. 131.

5.2.2 Eliminação de Produções ε

É possível transformar qualquer GIC numa outra GIC, "essencialmente equivalente" à original, mas que não contém produções ε.

Obviamente a palavra vazia só pode ser gerada por uma GIC se esta tiver pelo menos uma produção ε na sua lista de produções.

O método que vamos apresentar constrói uma gramática sem produções ε, equivalente à gramática original a menos da palavra vazia. Se necessário, uma simples transformação permite obter uma gramática exactamente equivalente à original, com uma única produção $S \to \varepsilon$, sendo que S não consta no lado direito das produções.

Definição 5.2.7 Variável Anulável.

Uma variável A é **anulável** quando $A \overset{\star}{\Rightarrow} \varepsilon$.

O algoritmo seguinte permite identificar as variáveis anuláveis de uma GIC.

Algoritmo 5.2.8 Determinação do Conjunto de Variáveis Anuláveis.

Entrada: uma GIC $G = (V, T, P, S)$.

Saída: o conjunto de variáveis anuláveis de G.

Para cada produção $A \to \varepsilon$

    Marcar a variável A;

Repetir

    Para cada variável A não marcada

        Para cada produção $A \to \alpha$

            Se todos os símbolos em α estão marcados;

            então marcar A;

até que nenhuma nova variável seja marcada.

As variáveis anuláveis são as variáveis marcadas.

Algoritmo 5.2.9 Eliminação de Produções ε.

Entrada: uma GIC G.

Saída: Uma gramática G' sem produções ε.

Determinar o conjunto das variáveis anuláveis de G;

Para cada produção $A \to X_1X_2 \cdots X_n$ de G

construir produções de G' $A \to \alpha_1\alpha_2 \cdots \alpha_n$, onde

$$\alpha_i = \begin{cases} X_i & \text{se } X_i \text{ não é anulável} \\ X_i \text{ ou } \varepsilon & \text{se } X_i \text{ é anulável;} \end{cases}$$

Eliminar todas as produções ε construídas.

Sempre que $\varepsilon \in \mathcal{L}(G)$, ou seja, quando S é anulável, o Algoritmo de Eliminação de Produções ε produz uma gramática G' tal que $\mathcal{L}(G') = \mathcal{L}(G) \setminus \{\varepsilon\}$.

Nesse caso, basta introduzir uma nova variável inicial, S_0, e adicionar as duas produções $S_0 \to \varepsilon$ e $S_0 \to S$ para obter uma gramática equivalente à original.

Nesta nova gramática, a produção $S_0 \to \varepsilon$ serve apenas para derivar a palavra vazia e não intervém na derivação de qualquer outra palavra, já que S_0 não aparece no lado direito das produções.

Exemplo 5.2.10

Consideremos a gramática G com produções:

$$\begin{aligned} S &\to AB \\ A &\to aAA \mid \varepsilon \\ B &\to bBB \mid \varepsilon \end{aligned}$$

As variáveis anuláveis são: A, B, S.

As produções da nova gramática G' são:

$$S \to AB \mid A \mid B$$
$$A \to aAA \mid aA \mid a$$
$$B \to bBB \mid bB \mid b$$

Assim, como S é anulável, $\mathcal{L}(G') = \mathcal{L}(G) \setminus \{\varepsilon\}$. Neste caso, atendendo a que S não aparece no lado direito das produções, basta juntar a produção $S \to \varepsilon$ a G' para que $\mathcal{L}(G) = \mathcal{L}(G')$.

5.2.3 Eliminação de Produções Unitárias

Vamos verificar que é possível eliminar de uma GIC todas as produções que apenas substituem uma variável por outra.

Definição 5.2.11 Produção Unitária.

Uma produção é **unitária** se tem a forma $A \to B$, com $A, B \in V$.

Definição 5.2.12 Par Unitário de Variáveis.

(A, B) é **um par unitário de variáveis** sempre que $A \overset{\star}{\Rightarrow} B$ usando *apenas* produções unitárias.

Deixamos como exercício o desenvolvimento de um algoritmo para determinar todos os pares unitários de variáveis, baseado no seguinte processo indutivo: se (A, B) é um par unitário e $B \to C$ é uma produção então (A, C) é também um par unitário.

Algoritmo 5.2.13 Eliminação de Produções Unitárias.

Entrada: uma GIC $G = (V, T, P, S)$.

Saída: Uma gramática $G' = (V, T, P', S)$ sem produções unitárias.

Determinar todos os pares unitários de variáveis em G;

Copiar para P' todas as produções não unitárias em P;

Para cada par unitário (A, B)

Para cada produção não unitária $B \to \alpha$ em P

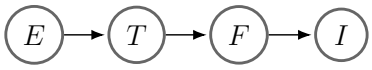
juntar a P' a produção $A \to \alpha$;

Exemplo 5.2.14

Consideremos a gramática $G = (\{E, T, F, I\}, \{0, 1, a, b, +, \times\}, P, E)$ com produções:

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

Os pares unitários de variáveis são dados pelos pares de vértices conexos do seguinte grafo de dependências unitárias:



As produções da nova gramática G' são:

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

5.2.4 A Forma Normal de Chomsky

Toda a gramática independente do contexto é redutível a uma gramática equivalente, na chamada forma normal de Chomsky, em que as derivações são essencialmente de dois tipos:

- A derivação de um terminal, em que uma variável é substituída por um terminal. Corresponde a uma ligação a uma folha na árvore de derivação.
- A derivação de duas variáveis, em que uma variável é substituída por duas. Corresponde a uma ramificação binária na árvore de derivação.

Definição 5.2.15 Forma normal de Chomsky.

Uma gramática independente do contexto, $G = (V, T, P, S)$, está na **forma normal de Chomsky** se cada uma das produções é de uma das duas seguintes formas:

1. $A \to a$ com $A \in V$ e $a \in T$;
2. $A \to BC$ com $A, B, C \in V$.

Em geral, uma gramática independente do contexto pode conter produções em que os lados direitos são sentenças com mais do que um símbolo terminal ou com uma mistura de símbolos terminais e variáveis.

Nesse caso, para reduzir uma gramática à forma normal de Chomsky é necessário substituir cada símbolo terminal a nos lados direitos dessas produções por uma nova variável X_a e definir uma produção adicional $X_a \to a$.

Definição 5.2.16 Terminal Não Isolado.

Um símbolo terminal a é **não isolado** se existe uma produção da forma $X \to \alpha a \beta$ em que $\alpha\beta \neq \varepsilon$, $\alpha, \beta \in (V \cup T)^\star$ e $X \in V$.

Deixamos como exercício o desenvolvimento de um algoritmo para identificar os símbolos terminais não isolados de uma GIC.

Algoritmo 5.2.17 Substituição de Símbolos Terminais.

Entrada: uma GIC $G = (V, T, P, S)$.

Saída: uma GIC $G' = (V', T, P', S)$ sem símbolos terminais não isolados.

Determinar o conjunto X dos símbolos terminais não isolados;

$V' \leftarrow V$;

$P' = \{A \to \alpha \in P : \ |\alpha| \leq 1\}$;

Para cada símbolo terminal a não isolado

definir uma nova variável associada ao símbolo a, X_a;

$V' \leftarrow V' \cup \{X_a\}$;

$P' \leftarrow P' \cup \{X_a \to a\}$;

Para cada produção $X \to \alpha \in P$ tal que $|\alpha| \geq 2$

determinar α' substituindo em α cada símbolo terminal a por X_a;

$P' \leftarrow P' \cup \{X \to \alpha'\}$;

Qualquer produção com mais de dois símbolos no lado direito é facilmente transformável numa sequência de produções equivalentes, cada uma com apenas dois símbolos no lado direito.

Algoritmo 5.2.18 Redução dos Lados Direitos a Dois Símbolos.

Substituir cada produção da forma $X \to X_1 X_2 \cdots X_k$, com $k \geq 3$, pelas produções

$$\begin{aligned} X &\to X_1 Y_1 \\ Y_1 &\to X_2 Y_2 \\ &\cdots \\ Y_{k-1} &\to X_{k-1} X_k \end{aligned}$$

onde $Y_1, Y_2, \ldots, Y_{k-1}$ são novas variáveis.

As diversas transformações vistas anteriormente podem ser combinadas em série por forma a simplificar uma GIC e ainda obter a forma normal de Chomsky.

Teorema 5.2.19

Para toda a linguagem independente do contexto, L, existe uma gramática independente do contexto na forma normal de Chomsky, G, tal que $\mathcal{L}(G) = L \setminus \{\varepsilon\}$.

Demonstração.
A prova consiste em verificar que as transformações na ordem dada pelo algoritmo seguinte preservam a equivalência entre gramáticas (a menos da palavra vazia).

Algoritmo 5.2.20 Redução à Forma Normal de Chomsky.

Dada uma GIC:

1. Eliminar os símbolos e produções inúteis;

2. Substituir os símbolos terminais que não ocorram isolados no lado direito das produções por variáveis;
3. Reduzir os lados direitos das produções a duas variáveis.
4. Eliminar as produções ε;
5. Eliminar as produções unitárias.

Exemplo 5.2.21

Vamos reduzir à forma normal de Chomsky a gramática de produções

$$\begin{aligned} S &\to ASB \mid \varepsilon \\ A &\to aAS \mid a \\ B &\to SbS \mid A \mid bb \end{aligned}$$

1. Eliminar os símbolos e produções inúteis: Todos os símbolos são geradores e atingíveis.
2. Substituir os símbolos terminais não isolados nos lados direitos das produções: Basta introduzir as variáveis X e Y e as produções $X \to a$ e $Y \to b$, obtendo

$$\begin{aligned} S &\to ASB \mid \varepsilon \\ A &\to XAS \mid a \\ B &\to SYS \mid A \mid YY \\ X &\to a \qquad Y \to b \end{aligned}$$

3. Reduzir os lados direitos das produções a duas variáveis: Para reduzir as sequências ASB, XAS e SYS, basta introduzir as variáveis C, D e E, obtendo

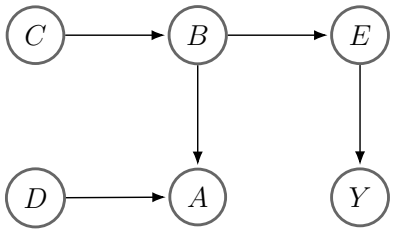
$$\begin{aligned} S &\to AC \mid \varepsilon \\ A &\to XD \mid a \\ B &\to SE \mid A \mid YY \\ C &\to SB \\ D &\to AS \\ E &\to YS \\ X &\to a \qquad Y \to b \end{aligned}$$

4. Eliminar as produções ε: Como o único símbolo anulável S nunca aparece isolado num lado direito das produções, basta eliminar a produção $S \to \varepsilon$

e detectar as ocorrências (anuláveis) de S nos lados direitos das produções, obtendo

$$
\begin{aligned}
&S \to AC \\
&A \to XD \mid a \\
&B \to SE \mid E \mid A \mid YY \\
&C \to SB \mid B \\
&D \to AS \mid A \\
&E \to YS \mid Y \\
&X \to a \qquad Y \to b
\end{aligned}
$$

5. Eliminar as produções unitárias: O grafo de dependências unitárias é o seguinte:



Por substituição dos pares unitários obtemos:

$$
\begin{aligned}
&S \to AC \\
&A \to XD \mid a \\
&B \to SE \mid YY \mid XD \mid YS \mid a \mid b \\
&C \to SB \mid XD \mid SE \mid YY \mid YS \mid a \mid b \\
&D \to AS \mid XD \mid a \\
&E \to YS \mid b \\
&X \to a \qquad Y \to b
\end{aligned}
$$

Finalmente, uma vez que a variável S é anulável, definimos uma nova variável inicial S_0 e juntamos as produções $S_0 \to \varepsilon$ e $S_0 \to S$, obtendo assim uma gramática na forma normal de Chomsky equivalente à gramática inicial.

Observações finais. O número de produções resultantes de uma redução à forma normal de Chomsky depende, entre outros factores, da ordem pela qual são realizadas as transformações indicadas no Algoritmo 5.2.20, p. 138.

Essas transformações não podem ser realizadas em ordem arbitrária, já que uma transformação pode invalidar o efeito da anterior. Por exemplo, não se pode exe-

cutar o algoritmo de eliminação de produções unitárias antes do algoritmo para eliminação de produções ε.

Se usarmos como medida do tamanho de uma gramática a soma dos tamanhos dos lados direitos de todas as produções, dependendo da ordem pela qual se efectuam as operações indicadas no Algoritmo 5.2.20, p. 138, é possível que o tamanho da gramática aumente exponencialmente ao efectuar a conversão.

Garantimos um aumento apenas quadrático, no pior caso, se efectuarmos a operação de redução do tamanho dos lados direitos das produções antes da operação de eliminação de produções ε, conforme indicamos no Algoritmo 5.2.20, p. 138.

Um exemplo ilustrativo deste fenómeno é dado pela conversão à forma normal de Chomsky da gramática G com produções $S \to A_1 A_2 \cdots A_n$ e $A_i \to a_i \mid \varepsilon$, $i = 1, \ldots, n$.

Se reduzirmos em primeiro lugar os lados direitos das produções a dois ou menos símbolos e só depois eliminarmos as produções ε obtemos uma gramática com tamanho da ordem de n^2. Quando eliminamos em primeiro lugar as produções ε, assistimos a um crescimento exponencial do número de produções e obtemos uma gramática com tamanho da ordem de 2^n.

5.3 Gramáticas e linguagens regulares

Na Secção 5.1, p. 123, referimos que a classe das linguagens regulares é uma subclasse própria da classe das linguagens independentes do contexto. Vamos agora verificar que as linguagens regulares podem ser geradas com um tipo simples de gramáticas independentes do contexto.

Começamos com a definição de gramática linear, na qual as produções têm no máximo uma variável em cada um dos seus lados direitos. Quando a variável no lado direito de cada uma das produções ocorre sempre no início da sentença (mais à esquerda) a gramática é linear à esquerda, quando ocorre no final da sentença (mais à direita) a gramática é linear à direita.

Definição 5.3.1 Gramática Linear.

Uma gramática independente do contexto $G = (V, T, P, S)$ é

- *linear* se todas as suas produções são da forma
  1. $X \to \alpha$ com $X \in V$ e $\alpha \in T^\star$,
  2. $X \to \beta Y \lambda$ com $X, Y \in V$ e $\beta, \lambda \in T^\star$.
- *linear à esquerda* se todas as suas produções são da forma
  1. $X \to \alpha$ com $\alpha \in T^\star$ e $X \in V$,
  2. $X \to Y \lambda$ com $X, Y \in V$ e $\lambda \in T^\star$.
- *linear à direita* se todas as suas produções são da forma
  1. $X \to \alpha$ com $X \in V$ e $\alpha \in T^\star$,
  2. $X \to \beta Y$ com $X, Y \in V$ e $\beta \in T^\star$.

Vamos demonstrar que as linguagens que são geradas por gramáticas lineares à esquerda são precisamente as mesmas que as geradas por gramáticas lineares à direita. Nesse sentido, começamos por definir uma subclasse de gramáticas lineares à direita.

Definição 5.3.2 Gramática Regular.

Uma gramática independente do contexto $G = (V, T, P, S)$ é regular se todas as suas produções são da forma:

1. $X \to aY$ com $a \in T$ e $X, Y \in V$,
2. $X \to \varepsilon$ com $X \in V$.

Resulta das definições que qualquer gramática regular é também linear à direita. Provamos agora que a maior generalidade das produções de uma gramática linear à direita, comparativamente com as produções de uma gramática regular, não acrescenta qualquer poder generativo.

Teorema 5.3.3

Toda a linguagem gerada por uma gramática linear à direita é também gerada por uma gramática regular.

Demonstração.

Basta reparar que:

1. qualquer produção da forma $X \to a_1 a_2 \cdots a_k Y$, com $k \geq 2$, pode ser

transformada na sequência de produções

$$X \to a_1 Y_1 \quad Y_1 \to a_2 Y_2 \quad \cdots \quad Y_{k-1} \to a_k Y$$

onde $Y_1, Y_2, \ldots Y_{k-1}$ são novas variáveis;

2. qualquer produção da forma $X \to a_1 a_2 \cdots a_k$, com $k \geq 2$, pode ser transformada na sequência de produções

$$X \to a_1 X_1 \quad X_1 \to a_2 X_2 \quad \cdots \quad X_{k-1} \to a_k X_k \quad X_k \to \varepsilon$$

onde $X_1, X_2, \ldots X_k$ são novas variáveis;

3. é possível eliminar as produções unitárias (ver o Algoritmo 5.2.13, p. 135).

Vamos agora associar a cada gramática regular um autómato finito equivalente, no sentido em que a linguagem gerada pela gramática é a mesma que a linguagem reconhecida pelo autómato.

Teorema 5.3.4

Para toda a gramática regular G existe um autómato finito A tal que $\mathcal{L}(A) = \mathcal{L}(G)$.

Demonstração.

Seja $G = (V, T, P, S)$ uma qualquer gramática regular. Especificamos o AFND $A = (Q, \Sigma, \delta, s, F)$ por:

- $Q = V$ (i.e., os estados do autómato são as variáveis da gramática);
- $\Sigma = T$ (i.e., o alfabeto do autómato é o conjunto dos símbolos terminais da gramática);
- $s = S$ (i.e., o estado inicial do autómato é axioma da gramática);
- $F = \{X \in V : \ X \to \varepsilon \in P\}$ (i.e., os estados de aceitação do autómato são as variáveis da gramática associadas a produções ε);
- para $X \in Q$ e $a \in \Sigma$, $\delta(X, a) = \{Y \in V : \ X \to aY \in P\}$ (i.e., existe uma transição de um estado X para um estado Y por um símbolo a se e só se a gramática contém uma produção $X \to aY$).

Usando indução sobre o tamanho das palavras $w \in \Sigma^+$, podemos provar que $Y \in \delta^\star(X, w)$ se e só se $X \overset{\star}{\Rightarrow} wY$. Com base nesta propriedade, demonstramos ainda que para qualquer palavra $w \in \Sigma^\star$, a transição $\delta^\star(s, w) \in F$ se e só se $S \overset{\star}{\Rightarrow} w$. Concluímos assim que a linguagem reconhecida pelo autómato é a mesma que a linguagem gerada pela gramática.

Teorema 5.3.5

Para todo o autómato finito A existe uma gramática regular G tal que $\mathcal{L}(A) = \mathcal{L}(G)$.

Demonstração.

Seja $A = (Q, \Sigma, \delta, s, F)$ um qualquer AFD. Especificamos a gramática regular $G = (V, T, P, S)$ por:

- $V = Q$ (i.e., as variáveis da gramática são os estados do autómato);
- $T = \Sigma$ (i.e., os símbolos terminais da gramática são os símbolos do alfabeto do autómato);
- $S = s$ (i.e., o axioma da gramática é o estado inicial do autómato);
- para $X, Y \in V$ e $a \in T$, $X \to aY \in P$ se e só se $\delta(X, a) = Y$ (i.e., a cada transição de um estado X para um estado Y por um símbolo a corresponde a produção $X \to aY$ na gramática);
- para cada $X \in F$, $X \to \varepsilon \in P$ (i.e., a cada estado de aceitação corresponde uma produção ε da gramática.

Deixamos como exercício os detalhes da demonstração de que a linguagem reconhecida por este autómato é a mesma que a linguagem gerada pela gramática.

Uma consequência imediata dos dois teoremas anteriores é que a classe das linguagens regulares é idêntica à classe das linguagens geradas por gramáticas regulares.

Corolário 5.3.6

Uma linguagem é regular se e só se é gerada por uma gramática regular.

Combinando este último resultado com o Teorema 5.3.3, p. 142 concluímos que a classe das linguagens regulares é idêntica à classe das linguagens lineares à direita.

Corolário 5.3.7

Uma linguagem é regular se e só se é gerada por uma gramática linear à direita.

A cada gramática linear à esquerda, $G = (V, T, P, S)$, podemos associar uma gramática $G^{-1} = (V, T, P^{-1}, S)$ em que as produções se obtêm das produções em P revertendo os lados direitos: para $X, Y \in V$ e $\alpha \in T^\star$, $X \to Y\alpha \in P$ se e só se $X \to \alpha^{-1}Y \in P^{-1}$ e $X \to \alpha \in P$ se e só se $X \to \alpha^{-1} \in P^{-1}$.

Claramente, G^{-1} é uma gramática linear à direita e as linguagens geradas pelas duas gramáticas são reversas uma da outra.

Uma conclusão semelhante é obtida se partirmos de uma gramática linear à di-

reita e revertermos os lados direitos das produções. Deixamos como um exercício simples de demonstração por indução que $\mathcal{L}(G^{-1}) = \mathcal{L}(G)^{-1}$.

É agora tarefa fácil provar que as classes das linguagens lineares à direita e à esquerda são idênticas.

Teorema 5.3.8

Uma linguagem é gerada por uma gramática linear à direita se e só se é gerada por uma gramática linear à esquerda.

Demonstração.

Se L é a linguagem gerada por uma gramática G, linear à esquerda então G^{-1} é uma gramática linear à direita que gera a linguagem L^{-1}.

Pelo Corolário 5.3.7, p. 144, L^{-1} é uma linguagem regular e como a classe das linguagens regulares é fechada para a operação de reversão, também $L = (L^{-1})^{-1}$ é uma linguagem regular. Logo, L é gerada por uma gramática linear à direita.

Seja agora L a linguagem gerada por uma gramática linear à direita. Pelo Corolário 5.3.7, p. 144, L é uma linguagem regular e como a classe das linguagens regulares é fechada para a operação de reversão, também L^{-1} é uma linguagem regular. Logo, L^{-1} é gerada por uma gramática G, linear à direita. Assim, G^{-1} é uma gramática linear à esquerda que gera $(L^{-1})^{-1} = L$.

Deixamos como exercício o desenvolvimento dos algoritmos que permitem converter uma gramática linear à direita numa gramática linear à esquerda e vice-versa (implícitos na demonstração do teorema anterior).

Exemplo 5.3.9

Seja $G = (\{S, X, Y, Z\}, \{a, b\}, P, S)$ a gramática linear à esquerda com produções

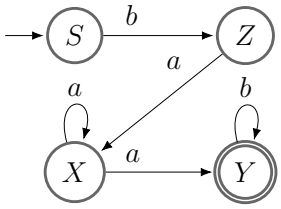
$$S \to Xab \quad X \to Xa \mid Ya \quad Y \to Yb \mid \varepsilon$$

A gramática $G^{-1} = (\{S, X, Y, Z\}, \{a, b\}, P^{-1}, S)$ com produções

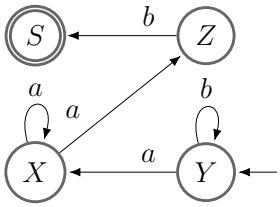
$$S \to baX \quad X \to aX \mid aY \quad Y \to bY \mid \varepsilon$$

é linear à direita e gera a linguagem $\mathcal{L}(G)^{-1}$.

Um AFND que reconhece a linguagem $\mathcal{L}(G)^{-1}$ é dado pelo seguinte diagrama de transições (introduzimos uma variável auxiliar Z para transformar a gramática numa gramática regular):



O autómato reverso deste autómato é o seguinte:



A linguagem reconhecida por este autómato é a linguagem reversa de $\mathcal{L}(G)^{-1}$, ou seja, a linguagem $\mathcal{L}(G)$. A este autómato corresponde a gramática linear à direita $G' = (\{S, X, Y, Z\}, \{a, b\}, P', Y)$ definida por

$$Y \to aX \mid bY \quad X \to aX \mid aZ \quad Z \to bS \quad S \to \varepsilon$$

Embora equivalentes, as gramáticas lineares à direita permitem a escrita de analisadores sintácticos mais simples do que aqueles que utilizam gramáticas lineares à esquerda.

Exemplo 5.3.10

Consideremos a gramática linear à esquerda G definida no Exemplo 5.3.9, p. 145.

Pretendemos averiguar se a palavra $aabbab$ é ou não derivável a partir de S. Para o decidir, é necessário analisar as produções X e as produções Y de modo a tentar encontrar uma derivação que comece com o prefixo $aabb$ da palavra dada, sendo este um processo complexo.

Por outro lado, usando a gramática linear à direita equivalente, G', vemos que a única produção possível de aplicar a partir do axioma Y é $Y \to aX$. Em seguida, apenas é possível aplicar $X \to aZ$, seguida de $Z \to bS$. Chegados a este ponto, a única produção que é possível aplicar é $S \to \varepsilon$, donde se conclui que a palavra não pertence à linguagem.

Apesar de a classe das linguagens lineares à esquerda ser idêntica à classe das linguagens lineares à direita, existem linguagens geradas por gramáticas lineares que não são regulares, conforme ilustrado no exemplo seguinte.

Exemplo 5.3.11

As linguagens $L_1 = \{a^n b^n c^k : \ n, k > 0\}$ e $L_2 = \{a^k b^n c^n : \ n, k > 0\}$ são geradas por gramáticas lineares. No entanto, nenhuma é regular, uma vez que $L_1 \cap L_2 = \{a^n b^n c^n : \ n > 0\}$ não é independente do contexto (confirmar com o Exemplo 5.5.3, p. 156).

5.4 Ambiguidade

Existem geralmente diversas possibilidades de derivar uma palavra numa GIC. No decorrer de um processo de derivação é por vezes possível usar duas ou mais produções distintas para substituir uma determinada variável presente numa sentença intermédia. Podem ainda surgir sentenças intermédias que contêm duas ou mais variáveis e a ordem pela qual se substituem essas variáveis origina diferentes sequências de derivações para uma mesma palavra.

Definição 5.4.1 Derivações à esquerda e à direita.

Uma **derivação à esquerda**, denotada por $\alpha \underset{\text{esq}}{\overset{\star}{\Rightarrow}} \beta$, é uma derivação de uma sentença β a partir de uma sentença α em que cada uma das sentenças intermédias resulta da substituição da variável mais à esquerda na sentença que a precede;

Uma **derivação à direita**, denotada por $\alpha \underset{\text{dir}}{\overset{\star}{\Rightarrow}} \beta$, é uma derivação de uma sentença β a partir de uma sentença α em que cada uma das sentenças intermédias resulta da substituição da variável mais à direita na sentença que a precede.

Exemplo 5.4.2

Com a gramática de expressões aritméticas considerada no Exemplo 5.1.8, p. 128, podemos estabelecer, entre outras, as seguintes derivações para a palavra $(a + b) \times 10$:

1. Seguindo sempre a derivação (mais) à esquerda, representada por

$$E \underset{\text{esq}}{\overset{\star}{\Rightarrow}} (a + b) \times 10$$

2. Seguindo sempre a derivação (mais) à direita, representada por

$$E \underset{\text{dir}}{\overset{\star}{\Rightarrow}} (a + b) \times 10.$$

Estas derivações são ilustradas em seguida:

$$E \underset{\text{esq}}{\overset{\star}{\Rightarrow}} (a+b)\times 10$$

$$\begin{aligned}
E &\Rightarrow E \times E \\
&\Rightarrow (E) \times E \\
&\Rightarrow (E+E) \times E \\
&\Rightarrow (I+E) \times E \\
&\Rightarrow (a+E) \times E \\
&\Rightarrow (a+I) \times E \\
&\Rightarrow (a+b) \times E \\
&\Rightarrow (a+b) \times N \\
&\Rightarrow (a+b) \times N0 \\
&\Rightarrow (a+b) \times 10
\end{aligned}$$

$$E \underset{\text{dir}}{\overset{\star}{\Rightarrow}} (a+b)\times 10$$

$$\begin{aligned}
E &\Rightarrow E \times E \\
&\Rightarrow E \times N \\
&\Rightarrow E \times N0 \\
&\Rightarrow E \times 10 \\
&\Rightarrow (E) \times 10 \\
&\Rightarrow (E+E) \times 10 \\
&\Rightarrow (E+I) \times 10 \\
&\Rightarrow (E+b) \times 10 \\
&\Rightarrow (I+b) \times 10 \\
&\Rightarrow (a+b) \times 10
\end{aligned}$$

Embora possam existir diversas formas de derivar uma palavra, se todas corresponderem à mesma "estrutura" de derivação então podem ser consideradas "equivalentes". A estrutura subjacente a uma derivação numa gramática independente ddo contexto é uma árvore ordenada com raiz, chamada árvore de derivação.

Definição 5.4.3 Árvore de derivação.

Uma **árvore de derivação** numa gramática $G = (V, T, P, S)$ é uma árvore ordenada, com raiz, tal que:

- a raiz é rotulada por S;
- cada nó interno é rotulado por uma variável de V;
- cada folha é rotulada por ε ou por um símbolo terminal de T;
- se um nó interno A tem como filhos os nós $(X_1, X_2, \ldots, X_n)$ então a produção $A \to X_1 X_2 \cdots X_n$ pertence ao conjunto de produções P.

O **valor** de uma árvore de derivação numa gramática é a palavra que resulta da concatenação dos símbolos terminais nas folhas da árvore numa travessia em pré-ordem (da esquerda para a direita). A **altura** de uma árvore é o comprimento (número de arestas) do caminho mais longo da raiz até às folhas.

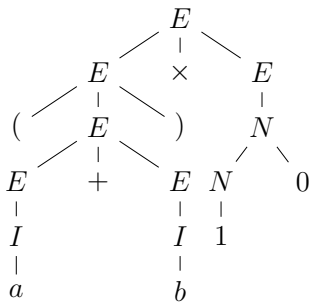
Designamos por **árvore de derivação parcial** uma qualquer sub-árvore de alguma árvore de derivação, com raiz rotulada por uma variável.

Exemplo 5.4.4

A árvore de derivação seguinte corresponde à derivação da palavra $(a + b) \times 10$ segundo a gramática do Exemplo 5.1.8, p. 128.

O valor da árvore de derivação de $(a + b) \times 10$ consiste na concatenação dos símbolos terminais nas folhas da árvore de derivação, caminhando da esquerda para a direita.

A estrutura da árvore não especifica a ordem pela qual se efectuaram as derivações e, neste caso, a árvore que se obtém é a mesma quer se sigam derivações à esquerda ou à direita.



A relação entre derivações e árvores de derivação é dada pelos resultados seguintes.

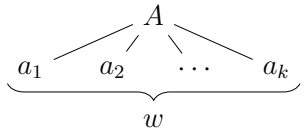
Teorema 5.4.5

Seja $G = (V, T, P, S)$ uma GIC. *$A \overset{\star}{\Rightarrow} w$ para alguma variável $A \in V$ e para alguma palavra $w \in T^\star$ se e só se existe uma árvore de derivação parcial com raiz A e valor w.*

Demonstração.

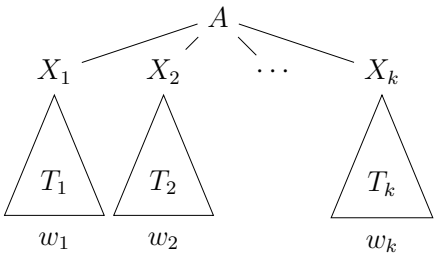
Suponhamos que $A \overset{\star}{\Rightarrow} w$ para alguma variável $A \in V$ e para alguma palavra $w \in T^\star$. Vamos demonstrar por indução sobre o número de passos da derivação que existe uma árvore de derivação parcial com raiz A e valor w.

Caso base. Se $A \Rightarrow w$ então $A \to w$ é uma produção da gramática e a seguinte árvore de derivação parcial tem valor w:



Passo indutivo. Estabelecemos como hipótese de indução que qualquer derivação em n ou menos passos possui uma árvore de derivação parcial. Consideremos uma derivação $A \overset{\star}{\Rightarrow} w$ em $n+1 \geq 2$ passos. Então $A \Rightarrow X_1 X_2 \cdots X_k \overset{\star}{\Rightarrow} w$, onde o primeiro passo corresponde à aplicação de alguma produção da forma $A \to X_1 X_2 \cdots X_k$, em que $X_i \in V \cup T$, $i = 1, \ldots, k$.

Seja $I = \{i : \ X_i \in V, i = 1, \ldots, k\}$. Para $i \in \{1, \ldots, k\} \setminus I$, X_i é um símbolo terminal, pelo que definimos $w_i = X_i$ e associamos a X_i uma árvore com um único nó $T_i = w_i$. Para $i \in I$, X_i é uma variável e portanto $X_i \overset{\star}{\Rightarrow} w_i$, com $w_i \in T^\star$, em n ou menos passos. Temos assim que $w = w_1 w_2 \cdots w_k$. Aplicando a hipótese de indução concluímos que existe uma árvore de derivação parcial T_i de raiz X_i e valor w_i, para cada $i \in I$. Assim, a árvore de derivação parcial a seguir ilustrada é uma árvore de derivação parcial de raiz A e valor w:



A demonstração da implicação contrária faz-se de forma semelhante, por indução sobre a altura da árvore de derivação.

Teorema 5.4.6

Existe uma e uma só derivação à esquerda $A \overset{\star}{\underset{\text{esq}}{\Rightarrow}} w$*, bem como uma e uma só derivação à direita* $A \overset{\star}{\underset{\text{dir}}{\Rightarrow}} w$*, associada a cada árvore de derivação parcial de raiz* A *e valor* w*.*

Demonstração.

Duas árvores de derivação distintas T_1 e T_2, com o mesmo valor w, estão associadas a duas derivações à esquerda distintas, já que a cada derivação, mesmo que não à esquerda, corresponde uma única árvore de derivação.

Por outro lado, sejam $S \overset{\star}{\underset{\text{esq}}{\Rightarrow}} \alpha A \beta \overset{\star}{\underset{\text{esq}}{\Rightarrow}} \alpha \gamma_1 \beta$ e $S \overset{\star}{\underset{\text{esq}}{\Rightarrow}} \alpha A \beta \overset{\star}{\underset{\text{esq}}{\Rightarrow}} \alpha \gamma_2 \beta$ duas derivações à esquerda distintas de uma mesma palavra, às quais correspondem duas árvores de derivação T_1 e T_2, respectivamente. Nestas duas expressões, $\alpha \in T^\star$ indica a parte inicial da palavra já derivada à esquerda e que é comum às duas derivações, A é a variável a partir da qual as derivações são diferentes e $\beta \in (V \cup T)^\star$ é a parte comum ainda por derivar.

Assim, na derivação a partir de A são aplicadas pelo menos duas produções distintas pelo que as respectivas sub-árvores parciais de raiz A são diferentes uma

vez que as duas sequências dos filhos de A são diferentes (lados direitos das produções). Assim, também as árvores de derivação são diferentes.

A demonstração da unicidade das derivações à direita é semelhante, pelo que se omite.

Notamos que o teorema anterior não exclui a possibilidade de duas derivações diferentes de uma mesma palavra corresponderem a árvores de derivação diferentes. Nesse caso, a derivação dessa palavra é ambígua.

Definição 5.4.7 Gramática Ambígua.

Uma gramática $G = (V, T, P, S)$ é **ambígua** quando existe pelo menos uma palavra $w \in T^\star$ para a qual existem duas árvores de derivação distintas.

Tendo presente os resultados anteriores, concluímos que uma gramática é ambígua se e só se existe alguma palavra com duas derivações à esquerda distintas.

Uma mesma linguagem pode ser gerada por diversas gramáticas, podendo umas ser ambíguas e outras não.

Definição 5.4.8 Linguagem Inerentemente Ambígua.

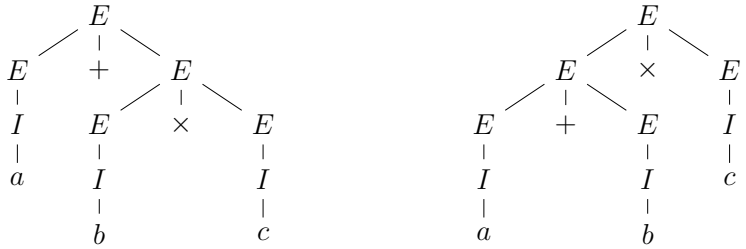
Uma linguagem é **inerentemente ambígua** quando todas as suas gramáticas geradoras são ambíguas.

Exemplo 5.4.9

Ainda sobre a gramática das Expressões Aritméticas, consideremos duas derivações distintas para $a + b \times c$:

$$
\begin{aligned}
E &\Rightarrow E + E \\
&\Rightarrow I + E \\
&\Rightarrow a + E \\
&\Rightarrow (I + E) \times E \\
&\Rightarrow a + E \times E \\
&\Rightarrow a + I \times E \\
&\Rightarrow a + b \times E \\
&\Rightarrow a + b \times I \\
&\Rightarrow a + b \times c
\end{aligned}
\qquad
\begin{aligned}
E &\Rightarrow E \times E \\
&\Rightarrow E + E \times E \\
&\Rightarrow I + E \times E \\
&\Rightarrow a + E \times E \\
&\Rightarrow a + I \times E \\
&\Rightarrow a + b \times E \\
&\Rightarrow a + b \times I \\
&\Rightarrow a + b \times c
\end{aligned}
$$

Mesmo seguindo sempre derivações à esquerda, são geradas duas árvores de derivação distintas. As árvores associadas às derivações anteriores são:



Tratando-se do cálculo do valor de uma expressão aritmética as duas derivações conduzem, efectivamente, a dois resultados diferentes!

Remoção da Ambiguidade. A ambiguidade na gramática das expressões aritméticas do Exemplo 5.1.8, p. 128 resulta do facto de não estipular as regras habituais de precedência dos operadores. É simples, neste caso, construir uma gramática equivalente que gera a mesma linguagem e que não é ambígua, conforme ilustramos no Exemplo 5.4.10, p. 152.

No entanto, nem sempre é possível remover a ambiguidade de uma linguagem. Por exemplo, prova-se que a linguagem independente do contexto

$$L = \{a^n b^n c^m d^m : \ n, m \geq 1\} \cup \{a^n b^m c^m d^n : \ n, m \geq 1\}$$

é inerentemente ambígua. De facto, uma palavra do tipo $a^k b^k c^k d^k$ terá sempre duas árvores de derivação distintas em qualquer gramática geradora de L.

Por outro lado, as linguagens regulares nunca são inerentemente ambíguas uma vez que podem ser especificadas por AFD e o reconhecimento de cada palavra corresponde a um único caminho, de comprimento igual ao tamanho da palavra, do estado inicial para o estado de aceitação.

Vimos na Secção 5.3, p. 141 que qualquer linguagem regular pode ser definida por uma GIC que contém apenas produções do tipo $A \to \varepsilon$ e $A \to aB$, com $X, Y \in V$ e $a \in T$. É imediato constatar que cada derivação usando este tipo de produções corresponde a uma única árvore que se reduz a um caminho (omitindo as folhas).

Exemplo 5.4.10

A forma de remover a ambiguidade na gramática das expressões aritméticas do Exemplo 5.1.8, p. 128 consiste em estabelecer a precedência do produto em relação à soma, introduzindo duas novas variáveis *termo* (T) e *factor* (F).

$$G' = (\{E, I, N, T, F\}, \{a, b, 0, 1, (,), +, \times, -\}, P', E)$$

onde P' é o conjunto de produções

$$E \to T \mid E + T$$

$$
\begin{aligned}
T &\to F \mid T \times F \\
F &\to I \mid N \mid (E) \\
I &\to a \mid b \mid Ia \mid Ib \\
N &\to 0 \mid 1 \mid N0 \mid N1 \mid -N \mid +N
\end{aligned}
$$

Exemplo 5.4.11

A instrução

$$\text{if } E_1 \text{ then if } E_2 \text{ then } S_1 \text{ else } S_2$$

possui duas árvores de derivação distintas com a gramática:

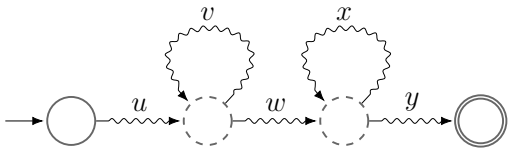
$$
\begin{aligned}
\langle \text{instr} \rangle \to\ & \text{if } \langle \text{expr} \rangle \text{ then } \langle \text{instr} \rangle \\
& \mid \text{if } \langle \text{expr} \rangle \text{ then } \langle \text{instr} \rangle \text{ else } \langle \text{instr} \rangle \\
& \mid \langle \text{other} \rangle
\end{aligned}
$$

A mesma instrução possui apenas uma árvore de derivação com a gramática:

$$
\begin{aligned}
\langle \text{instr} \rangle \to\ & \langle \text{closed-instr} \rangle \mid \langle \text{open-instr} \rangle \\
\langle \text{open-instr} \rangle \to\ & \text{if } \langle \text{expr} \rangle \text{ then } \langle \text{instr} \rangle \\
& \mid \text{if } \langle \text{expr} \rangle \text{ then } \langle \text{closed-instr} \rangle \text{ else } \langle \text{open-instr} \rangle \\
\langle \text{closed-instr} \rangle \to\ & \text{if } \langle \text{expr} \rangle \text{ then } \langle \text{closed-instr} \rangle \text{ else } \langle \text{closed-instr} \rangle \\
& \mid \langle \text{other} \rangle
\end{aligned}
$$

5.5 O lema da bombagem para LIC

O lema da bombagem indica que qualquer palavra suficientemente longa de uma LIC pode ser dividida em cinco partes tais que: toda a palavra obtida da palavra original por substituição da segunda e da quarta partes por um número igual de repetições das respetivas partes, continua a pertencer à linguagem.



Na demonstração do lema da bombagem, vamos considerar árvores de derivação numa gramática na forma normal de Chomsky nas quais ignoramos as derivações dos símbolos terminais. Cada uma destas árvores é uma **árvore estritamente binária** em que os nós têm 2 filhos (nós internos) ou 0 filhos (folhas). Todos os

nós são ainda rotulados por variáveis. São válidas as seguintes relações entre o número de folhas e a altura de uma árvore estritamente binária:

1. O número de folhas f de uma árvore com altura h satisfaz $h + 1 \leq f \leq 2^h$.
2. A altura h de uma árvore com 2^m folhas satisfaz, $m \leq h \leq 2^m - 1$.

Lema 5.5.1 Lema da Bombagem para LIC.

Se L é uma linguagem independente do contexto sobre um alfabeto Σ então existe um inteiro positivo n tal que, para qualquer palavra $z \in L$ de tamanho $|z| \geq n$, existem $u, v, w, x, y \in \Sigma^\star$ tais que:

1. $z = uvwxy$;
2. $|vwx| \leq n$;
3. $|vx| > 0$;
4. $\forall k \geq 0,\ uv^k wx^k y \in L$.

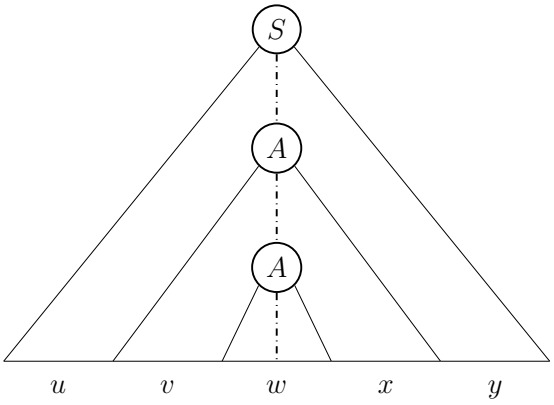
Demonstração.

Sejam L uma linguagem independente do contexto e G uma gramática na forma normal de Chomsky tais que $\mathcal{L}(G) = L \setminus \{\varepsilon\}$.

Seja m o número de variáveis de G. A qualquer palavra $z \in L$ de tamanho $|z| \geq n = 2^m$ está associada uma árvore de derivação estritamente binária de altura maior ou igual a m com nós rotulados por variáveis. Assim, no caminho mais longo da raiz até às folhas existem pelo menos $m + 1$ nós rotulados por variáveis.

Uma vez que existem apenas m variáveis, no caminho mais longo da raiz até às folhas existem pelo menos dois nós rotulados com uma mesma variável A:

$$S \overset{\star}{\Rightarrow} u\ A\ y$$
$$A \overset{\star}{\Rightarrow} v\ A\ x = vwx$$
$$A \overset{\star}{\Rightarrow} w$$



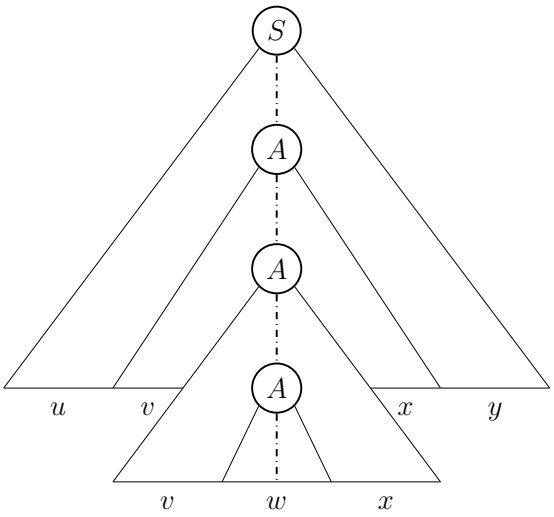
Atendendo a que $A \overset{\star}{\Rightarrow} vAx$ e que a gramática não tem produções ε, concluímos que $|vx| > 0$.

Sejam r e s dois nós rotulados com a mesma variável A e situados o mais profundamente possível na árvore, isto é, s é descendente de r e os caminhos da raiz até às folhas em cada uma das sub-árvores esquerda e direita de r são de nós rotulados com variáveis diferentes de A.

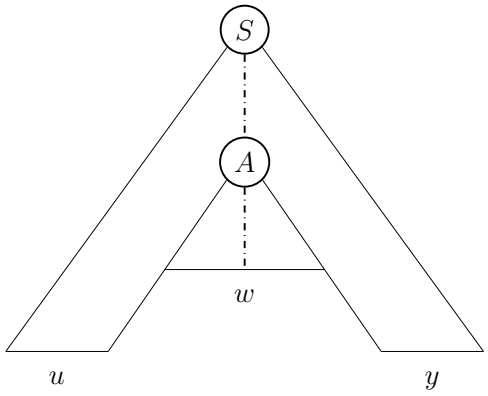
Nestas condições, concluímos que $|vwx| \leq 2^m = n$ e, uma vez que $A \overset{\star}{\Rightarrow} vAx$, podemos ir derivando sucessivamente,

$$S \overset{\star}{\Rightarrow} uAy \overset{\star}{\Rightarrow} uvAxy \overset{\star}{\Rightarrow} uvvAxxy \overset{\star}{\Rightarrow} \cdots \overset{\star}{\Rightarrow} uv^kAx^ky.$$

No final, uma vez que $A \overset{\star}{\Rightarrow} w$, obtemos $S \overset{\star}{\Rightarrow} uv^kwx^ky$.



Mesmo para $k = 0$, $S \overset{\star}{\Rightarrow} uAy$ e como $A \overset{\star}{\Rightarrow} w$ também $S \overset{\star}{\Rightarrow} uwy$.



Se conseguirmos provar que uma determinada linguagem não satisfaz o lema anterior então podemos concluir que não é independente do contexto.

Teorema 5.5.2

Se uma linguagem não satisfaz o lema da bombagem então não é independente do contexto.

Exemplo 5.5.3

Suponhamos que a linguagem $L = \{a^m b^m c^m : \ m \geq 0\}$ é independente do contexto. Pelo lema da bombagem, existe um inteiro $n \in \mathbb{N}$ para o qual toda a palavra $z \in L$ com $|z| \geq n$ pode ser decomposta em $z = uvwxy$ de tal modo que: (1) $|vwx| \leq n$; (2) $|vx| > 0$; (3) $\forall k \geq 0, \ uv^k wx^k y \in L$.

Consideremos a palavra $z = a^n b^n c^n \in L$. Como z é suficientemente longa, existem palavras u, v, w, x e y tais que $z = uvwxy$.

Ora, para que se verifique $|vwx| \leq n$, a palavra vwx não pode conter a´s e c´s.

Suponhamos que vwx não contém c´s (e portanto y contém n c´s). Para que se verifique $|vx| > 0$, v ou x terão de conter pelo menos uma letra (um a ou um b). Mas então a palavra $uv^0wx^0y = uwy$ terá menos a´s ou b´s do que c´s, já que o número de c´s se mantém e o número de a´s ou b´s diminui. Assim, esta palavra não pertence à linguagem.

Caso vwx não contenha a´s, concluímos de forma semelhante que $uv^0wx^0y = uwy$ não pertence à linguagem.

Em qualquer dos casos chegamos a uma contradição com a terceira condição, pelo que L não pode ser uma linguagem independente do contexto.

Exemplo 5.5.4

Admitamos que $L = \{a^{2^n} : \ n \geq 0\}$ é independente do contexto. Seja $n \geq 1$ o parâmetro especificado no lema da bombagem e seja $z = a^{2^n} \in L$.

Consideremos uma qualquer partição $z = uvwxy$ tal que (i) $|vwx| \leq n$ e (ii) $|vx| \geq 1$. Pelo lema da bombagem, (iii) uv^kwx^ky é uma palavra da linguagem L, qualquer que seja $k \geq 0$.

Temos que $v = a^p$ e $x = a^q$, com $1 \leq p + q \leq n$ (por (i) e (ii)).

Para $k = 2$, por (iii), concluímos que $uv^2wx^2y \in L$. Mas $uv^2wx^2y = a^{2^n+p+q}$. No entanto, $2^n < 2^n + p + q \leq 2^n + n < 2^{n+1}$, pelo que $uv^2wx^2y \notin L$ o que é uma contradição. Concluímos que L não pode ser independente do contexto.

Mostrar que uma linguagem não é independente do contexto usando o lema da bombagem não é, muitas vezes, uma tarefa simples. Nesses casos, a aplicação de certas propriedades das LIC, reduzindo uma linguagem a uma ou mais linguagens que sabemos serem independentes do contexto, constitui uma alternativa mais viável, cujo estudo deixamos para o final do próximo capítulo.

5.6 Exercícios

1. Determine gramáticas independentes do contexto geradoras das linguagens a seguir definidas, argumentando de modo semi-formal sobre a correcção das soluções encontradas:

(a) a^*;

(b) a^+;

(c) $\{a^nb^n : \ n \geq 0\}$;

(d) $\{a^nb^n : \ n > 0\}$;

(e) $\{a^nb^{n+1} : \ n \geq 0\}$;

(f) $\{a^mb^n : \ 0 \leq m \leq n\}$;

(g) $\{a^mb^n : \ m \geq n \geq 0\}$;

(h) $\{a^mb^n : \ 0 \leq m \leq n + 3, \ n \geq 0\}$;

(i) $\{a^nb^{2n} : \ n \geq 0\}$;

(j) $\{a^mb^n : \ m \leq n \leq 2m, \ m \geq 0\}$;

(k) $\{a^mb^n : \ m < n < 2m, \ n \geq 0\}$;

(l) $\{a^mb^n : \ 2n \leq m \leq 3n, \ n \geq 0\}$;

(m) $\{a^m b^n : \ m \neq n, \ m, n \geq 0\}$;

(n) $\{a^m b^n c^p : \ m = n \text{ ou } n = p, \ m, n, p \geq 0\}$;

(o) $\{a^m b^n c^p : \ m \neq n \text{ ou } n \neq p, \ m, n, p \geq 0\}$;

(p) $\{a^m b^n a^n b^m : \ m, n \geq 0\}$;

(q) $\{a^m b^m a^n b^n : \ m, n \geq 0\}$;

(r) $\{a^m b^n c^p : \ m = n \text{ ou } n \leq p, \ m, n, p \geq 0\}$;

(s) $\{a^m b^n c^p : \ |m - n| = p, \ m, n, p \geq 0\}$;

2. Considere a linguagem $L = \{a^j b^{i+j} c^i : \ i, j \in \mathbb{N}_0\}$.

(a) Use o lema da bombagem para mostrar que L não é uma linguagem regular.

(b) Especifique uma gramática independente do contexto, G, geradora de L.

(c) Prove que $L = \mathcal{L}(G)$.

3. Considere a gramática ambígua

$$G = (\{S\}, \{a, b\}, \{S \to aS \mid aSbS \mid \varepsilon\}, S)\,.$$

(a) Para a palavra aab determine:

(a) duas árvores de derivação distintas;

(b) duas derivações à esquerda distintas;

(c) duas derivações à direita distintas.

(b) Construa uma gramática não ambígua equivalente.

4. Construa uma gramática independente do contexto para gerar todas as expressões regulares sobre o alfabeto $\{a, b\}$.

5. Determine se cada uma das linguagens seguintes é regular ou se é não regular mas independente do contexto. Justifique as respostas.

(a) $L_1 = \{(11)^m (00)^n : \ m, n \geq 0\}$;

(b) $L_2 = \{(11)^n (00)^n : \ n \geq 0\}$;

(c) $L_3 = \{(11)^m (00)^n : \ m \neq n, \ m, n \geq 0\}$;

(d) $L_4 = \{(11)^n (00)^n : \ 0 \leq n \leq 100\}$;

(e) $L_5 = \{(11)^m (00)^m (11)^n (00)^n : \ m, n \geq 0\}$;

(f) $L_6 = \{(11)^m (00)^n (11)^n (00)^m : \ m, n \geq 0\}$.

6. Considere a gramática $G = (\{S, A, B\}, \{0, 1\}, P, S)$ com as seguintes produ-

ções:

$$S \to 0AB \qquad A \to 0A \mid \varepsilon \qquad B \to 0B \mid 1B \mid \varepsilon.$$

(a) Construa derivações à esquerda e à direita para 0010 e as respectivas árvores de derivação

(b) Será esta gramática ambígua?

(c) Verifique que a linguagem gerada por esta gramática é regular.

(d) Determine uma gramática regular equivalente.

7. Considere a seguinte versão simplificada de uma gramática ambígua de expressões aritméticas na notação *infix* $G = (\{E\}, \{a, b, c, +, *\}, P, E)$, com as produções:

$$E \to E + E \mid E * E \mid a \mid b \mid c.$$

(a) Verifique que existe ambiguidade no cálculo de $a + b * c$.

(b) Construa uma gramática não ambígua equivalente.

(c) Construa agora uma gramática para a linguagem em que as mesma operações são escritas na notação sufixa (também conhecida por notação Polaca ou *postfix*) e mostre que essa gramática não é ambígua.

8. Mostre, usando um contra-exemplo, que é fundamental seguir a ordem indicada no Algoritmo 5.2.2, p. 131 para eliminação de símbolos inúteis.

9. Considere as possíveis árvores de derivação numa gramática na forma normal de Chomsky.

Note que numa árvore de derivação para uma gramática na forma normal de Chomsky cada folha é filho único do nó pai. Assim, existem tantas folhas como pais das folhas. Assim, se removermos as folhas obtemos uma árvore, com menos um nível do que a original, que é estritamente binária, isto é, cada nó tem 0 filhos ou 2 filhos. Para além disso, esta árvore tem tantas folhas como a original.

Demonstre as afirmações seguintes:

(a) Se uma árvore de derivação tem n níveis então o número de folhas (rotuladas por variáveis), f, satisfaz $n - 1 \leq f \leq 2^{n-2}$.

(b) Se uma árvore de derivação tem 2^m folhas então o número de nós num caminho mais longo da raiz até às folhas, $C_{\max}$, satisfaz $m+1 \leq C_{\max} \leq 2^m$.

10. Construa uma gramática independente do contexto para a linguagem

$$L = \{a^n w w^{-1} b^n : \ w \in \{a, b\}^\star, n \geq 1\}$$

e, em seguida, converta essa gramática à forma normal de Chomsky.

11. Considere a GIC $G = (\{S, X, Y, Z, W\}, \{a, b, c\}, P, S)$, com conjunto de produções P definido por

$$S \to XbS \mid YaS \mid ZcS \mid \varepsilon \qquad W \to Sa \mid a \mid b \mid c$$
$$X \to aS \mid bZ \qquad Y \to bS \mid aZ \qquad Z \to XYZ.$$

(a) Indique duas derivações distintas da palavra $abba$ com a gramática G.

(b) Comente a afirmação "Se uma palavra tiver duas derivações diferentes numa GIC então podemos concluir que é ambígua. Logo, G é ambígua."

(c) Construa uma GIC na forma normal de Chomsky geradora de $\mathcal{L}(G) \setminus \{\varepsilon\}$.

(d) Mostre (por indução) que para $n \in \mathbb{N}$, $a^n b^n \in \mathcal{L}(G)$.

(e) Identifique a linguagem $\mathcal{L}(G)$.

12. Usando o lema da bombagem para linguagens independentes do contexto, mostre que qualquer linguagem finita é independente do contexto.

13. Usando indução estrutural, mostre que as linguagens associadas às expressões regulares são geradas por gramáticas independentes do contexto.

14. Considere o contexto da demonstração do Teorema 5.3.4, p. 143. Verifique que, para qualquer palavra $w \in \Sigma^+$, $Y \in \delta^\star(X, w)$ se e só se $X \overset{\star}{\Rightarrow} wY$. Em seguida, prove que, para qualquer $w \in \Sigma^\star$, $\delta^\star(s, w) \in F$ se e só se $S \overset{\star}{\Rightarrow} w$.

15. Obtenha gramáticas lineares geradoras das seguintes linguagens:

(a) $L_1 = \{a^n b^n c^k : \ n, k > 0\}$.

(b) $L_2 = \{a^k b^n c^n : \ n, k \geq 0\}$.

16. Considere a LIC $L = \{a^n b^n : \ n \geq 0\}$.

(a) Construa uma GIC G geradora de L.

(b) Mostre que, para $k \geq 1$, L^k é uma LIC.

(c) Mostre que $L^\star$ é uma LIC.

(d) Mostre que $\mathcal{L}(a^* b^*) \setminus L$ é uma LIC.

17. Aplique o Lema da Bombagem para LIC, p. 154 para provar que as seguintes linguagens não são independentes do contexto:

(a) $L = \{ww : \ w \in \{a, b\}^\star\}$.

(b) $L = \{a^i b^j c^{ij} : \ i, j \in \mathbb{N}_0\}$.

(c) $L = \{a^n b^n a^n b^n : \ n \in \mathbb{N}_0\}$.

(d) $L = \{a^{n^2} : \ n \geq 1\}$.

(e) $L = \{a^p : \ p \text{ é um número primo}\}$.

(f) $L = \{w \in \{a,b,c\}^\star : \ \#_a(w) < \#_b(w) < \#_c(w)\}$.

(g) $L = \{w \in \{a,b,c\}^\star : \ \#_b(w) > \#_a(w)$ e $\#_c(w) > \#_a(w)\}$.

18. Determine gramáticas geradoras das seguintes linguagens independentes do contexto:

(a) $L = \{w \in \{a,b\}^\star : \ \#_a(w) = \#_b(w)\}$.

(b) $L = \{w \in \{a,b\}^\star : \ \#_a(w) = 2\#_b(w)\}$.

(c) $L = \{w \in \{a,b\}^\star : \ \#_a(w) < \#_b(w)\}$.

(d) $L = \{w \in \{a,b\}^\star : \ \#_a(w) \leq \#_b(w)\}$.

(e) $L = \{w \in \{a,b\}^\star : \ \#_a(w) \neq \#_b(w)\}$.

(f) $L = \{z \in \{a,b\}^\star :$ não existe $w \in \{a,b\}^\star$ tal que $z = ww\}$.

19. Existe um erro na seguinte demonstração de que a linguagem

$$L = \{w \in \{a,b\}^\star : \ \#_a(w) < \#_b(w) < 2\#_a(w)\}$$

não é independente do contexto. Descubra-o.

Demonstração: Seja $n \in \mathbb{N}$ o parâmetro especificado no lema da bombagem que garante que qualquer palavra de tamanho maior ou igual do que n pode ser bombeada.

Consideremos a palavra $z = a^{n+1}b^{n+2} \in L$, de tamanho $|z| = 2n+3 \geq n$.

Seja $z = uvwxy$ uma qualquer partição da palavra z nas condições do lema. Há 3 casos a considerar:

Caso 1: Se vwx só tem a´s, seja $p = \#_a(vx) \geq 1$. Então $\#_a(uv^kwx^ky) = n+1+p(k-1)$ e fixando $k = 2$ concluímos que $\#_a(uv^kwx^ky) \geq n+2 = \#_b(uv^kwx^ky)$.

Caso 2: Se vwx só tem b´s, seja $p = \#_b(vx) \geq 1$. Então $\#_b(uv^kwx^ky) = n+2+p(k-1)$ e fixando $k = 0$ concluímos que $\#_b(uv^kwx^ky) \leq n+1 = \#_a(uv^kwx^ky)$.

Caso 3: Se vwx contém a´s e b´s, sejam $p = \#_a(vx) \geq 1$ e $q = \#_b(vx) \geq 1$. Então $\#_a(uv^kwx^ky) = n+1+p(k-1)$ e $\#_b(uv^kwx^ky) = n+2+q(k-1)$.

Aqui temos duas possibilidades:

(a) Se $p \leq q$ então, fixando $k = 0$, concluímos que $\#_a(uv^kwx^ky) \geq \#_b(uv^kwx^ky)$.

(b) Se $p > q$ então, fixando $k = 2$, concluímos que $\#_a(uv^kwx^ky) \geq \#_b(uv^kwx^ky)$.

Em qualquer dos casos, existe um valor $k \in \mathbb{N}_0$ tal que $uv^kwx^ky \notin L$. Concluímos que L não satisfaz o lema da bombagem e, consequentemente, não é independente do contexto.

20. Considere a GIC G com as seguintes produções:

$$S \to ASA \mid aB \qquad A \to B \mid S \qquad B \to b \mid \varepsilon.$$

(a) Obtenha uma gramática equivalente, G', na forma normal de Chomsky.

(b) Seja $w \in \mathcal{L}(G')$ uma qualquer palavra de tamanho $n = |w|$. Determine, como função de n, o número de passos da derivação de w usando a gramática G'.

21. Mostre que a GIC G definida pelas seguintes produções é ambígua:

$$S \to aSbS \mid bSaS \mid \varepsilon.$$

Em seguida, determine uma gramática não ambígua geradora de $\mathcal{L}(G)$.