Capítulo 4

Lenguajes y gramáticas independientes del contexto

Como se ha visto, los autómatas son dispositivos que procesan cadenas de entrada. En capítulos posteriores consideraremos modelos de autómatas con mayor poder computacional que el de los modelos AFD-λ, AFN y AFD. En el presente capítulo iniciaremos el estudio de las **gramáticas generativas**, que son mecanismos para generar cadenas a partir de un símbolo inicial (el estudio de gramáticas continuará en el Capítulo 8).

☞ Los autómatas *procesan* cadenas

☞ Las gramáticas *generan* cadenas

Las gramáticas generativas y, en particular, las gramáticas independientes del contexto (GIC), fueron introducidas por Noam Chomsky en 1956 como un modelo para la descripción de los lenguajes naturales (como el español, el inglés, etc). En la década de los sesenta se comenzaron a usar GIC para presentar la sintaxis de lenguajes de programación y para el diseño de analizadores sintácticos en compiladores.

4.1. Gramáticas independientes del contexto

Una **gramática independiente del contexto** (GIC), también llamada **gramática contextual**, es una cuádrupla, $G = (V, \Sigma, S, P)$ formada por:

1. Un alfabeto V cuyos elementos se llaman **variables** o **símbolos no-terminales**.

2. Un alfabeto Σ cuyos elementos se llaman **símbolos terminales**. Se exige que los alfabetos Σ y V sean disyuntos.

3. Una variable especial $S \in V$, llamada **símbolo inicial** de la gramática.

4. Un conjunto finito $P \subseteq V \times (V \cup \Sigma)^*$ de **producciones** o **reglas de re-escritura**. Una producción $(A, w) \in P$ de G se denota por $A \to w$ y se lee "A produce w"; su significado es: la variable A se puede reemplazar (sobre-escribir) por la cadena w.

Notación y definiciones. Las variables se denotan con letras mayúsculas $A, B, C, \ldots$ Los elementos de Σ o símbolos terminales se denotan con letras minúsculas $a, b, c, \ldots$. Si $u, v \in (V \cup \Sigma)^*$ y $A \to w$ es una producción, se dice que uwv se **deriva directamente** de uAv, lo cual se denota por

$$uAv \Rightarrow uwv.$$

Si se quiere hacer referencia a la gramática G, se escribe

$$uAv \overset{G}{\Longrightarrow} uwv \quad \text{ó} \quad uAv \Longrightarrow_G uwv.$$

Si $u_1, u_2, \ldots, u_n$ son cadenas en $(V \cup \Sigma)^*$ y hay una sucesión de derivaciones directas

$$u_1 \overset{G}{\Longrightarrow} u_2, \quad u_2 \overset{G}{\Longrightarrow} u_3, \ \ldots \ , u_{n-1} \overset{G}{\Longrightarrow} u_n$$

se dice que u_n se deriva de u_1 y se escribe $u_1 \overset{*}{\Rightarrow} u_n$. La anterior sucesión de derivaciones directas se representa como

$$u_1 \Rightarrow u_2 \Rightarrow u_3 \Rightarrow \cdots \Rightarrow u_{n-1} \Rightarrow u_n$$

y se dice que es una **derivación** o una **generación** de u_n a partir de u_1. Para toda cadena w se asume que $w \overset{*}{\Rightarrow} w$; por lo tanto, $u \overset{*}{\Rightarrow} v$ significa que v se obtiene de u utilizando cero, una o más producciones de la gramática. Análogamente, $u \overset{+}{\Longrightarrow} v$ significa que v se obtiene de u utilizando una o más producciones.

El **lenguaje generado por una gramática** G se denota por $L(G)$ y se define como

$$L(G) := \{w \in \Sigma^* : S \overset{+}{\Longrightarrow} w\}.$$

Un lenguaje L sobre un alfabeto Σ se dice que es un **lenguaje independiente del contexto** (LIC) si existe una GIC G tal que $L(G) = L$. Dos GIC G_1 y G_2 son **equivalentes** si $L(G_1) = L(G_2)$.

La denominación "independiente del contexto" proviene del hecho de cada producción o regla de re-escritura $A \to w$ se aplica a la variable A independientemente de los caracteres que la rodean, es decir, independientemente del contexto en el que aparece A.

Ejemplo Sea G una gramática (V, Σ, S, P) dada por:

$$\begin{aligned}
&V = \{S\} \\
&\Sigma = \{a\} \\
&P = \{S \to aS,\ S \to \lambda\}
\end{aligned}$$

Se tiene $S \Rightarrow \lambda$ y

$$S \Rightarrow aS \overset{*}{\Rightarrow} a \cdots aS \Rightarrow a \cdots a.$$

Por consiguiente, $L(G) = a^*$.

De manera más simple, podemos presentar una gramática GIC listando sus producciones y separando con el símbolo $\mid$ las producciones de una misma variable. Se supone siempre que las letras mayúsculas representan variables y las letras minúsculas representan símbolos terminales. Así la gramática del ejemplo anterior se presenta simplemente como:

$$S \to aS \mid \lambda.$$

Ejemplo La gramática $G = (V, \Sigma, S, P)$ dada por:

$$\begin{aligned}
&V = \{S, A\} \\
&\Sigma = \{a, b\} \\
&P = \{S \to aS,\ S \to bA,\ S \to \lambda,\ A \to bA,\ A \to b,\ A \to \lambda\}
\end{aligned}$$

se puede presentar como

$$\begin{cases} S \to aS \mid bA \mid \lambda \\ A \to bA \mid b \mid \lambda \end{cases}$$

Se tiene $S \Rightarrow \lambda$. Todas las demás derivaciones en G comienzan ya sea con la producción $S \to aS$ o con $S \to bA$. Por lo tanto, tenemos

$$\begin{aligned}
&S \Rightarrow aS \overset{*}{\Rightarrow} a \cdots aS \Rightarrow a \cdots a. \\
&S \Rightarrow bA \overset{*}{\Rightarrow} b \cdots bA \Rightarrow b \cdots b. \\
&S \Rightarrow aS \overset{*}{\Rightarrow} a \cdots aS \Rightarrow a \cdots abA \overset{*}{\Rightarrow} a \cdots ab \cdots bA \Rightarrow a \cdots ab \cdots b.
\end{aligned}$$

Por consiguiente $L(G) = a^*b^*$.

Las siguientes gramáticas también generan el lenguaje a^*b^* y son, por lo tanto, equivalentes a G:

$$\begin{cases} S \to AB \\ A \to aA \mid \lambda \\ B \to bB \mid \lambda \end{cases} \qquad \begin{cases} S \to AB \mid \lambda \\ A \to aA \mid a \mid \lambda \\ B \to bB \mid b \mid \lambda \end{cases} \qquad \begin{cases} S \to aS \mid A \\ A \to bA \mid \lambda \end{cases}$$

Ejemplo La gramática

$$\begin{cases} S \to aS \mid aA \\ A \to bA \mid b \end{cases}$$

genera el lenguaje a^+b^+. Otra gramática equivalente es:

$$\begin{cases} S \to AB \\ A \to aA \mid a \\ B \to bB \mid b \end{cases}$$

Ejemplo La gramática

$$\begin{cases} S \to 1A \mid 0 \\ A \to 0A \mid 1A \mid \lambda \end{cases}$$

genera el lenguaje de los números naturales en numeración binaria. Nótese que la única cadena que comienza con 0, generable con esta gramática, es la cadena 0.

Ejemplo Encontrar una GIC que genere el lenguaje $0^*10^*10^*$ sobre $\Sigma = \{0, 1\}$, es decir, el lenguaje de todas las cadenas con exactamente dos unos.

Solución:

$$G: \quad \begin{cases} S \to A1A1A \\ A \to 0A \mid \lambda \end{cases}$$

Una gramática equivalente es

$$\begin{cases} S \to 0S \mid 1A \\ A \to 0A \mid 1B \\ B \to 0B \mid \lambda \end{cases}$$

Ejemplo Encontrar una GIC que genere el lenguaje $L = \{a^i b^i : i \geq 0\}$ sobre $\Sigma = \{a, b\}$, el cual no es un lenguaje regular.

Solución:

$$S \to aSb \mid \lambda.$$

Ejemplo Encontrar una GIC que genere el lenguaje de todos los palíndromos sobre $\Sigma = \{a, b\}$, el cual no es lenguaje regular.

Solución:

$$S \to aSa \mid bSb \mid a \mid b \mid \lambda.$$

Ejemplo Encontrar una GIC que genere el lenguaje de todas las cadenas sobre $\Sigma = \{a, b\}$ que tienen un número par de símbolos.

Solución:

$$S \to aSa \mid bSb \mid aSb \mid bSa \mid \lambda$$

Dos gramáticas equivalentes son:

$$\begin{cases} S \to aaS \mid bbS \mid abS \mid baS \mid \lambda \end{cases} \qquad\qquad \begin{cases} S \to AAS \mid \lambda \\ A \to a \mid b \end{cases}$$

Ejemplo Encontrar una GIC que genere el lenguaje $(ab \cup ba)^*$ sobre $\Sigma = \{a, b\}$.

Solución:

$$S \to abS \mid baS \mid \lambda.$$

Ejemplo Demostrar que la gramática G dada por:

$$S \to \texttt{(}S\texttt{)}S \mid \lambda$$

genera el lenguaje de todas las cadenas de paréntesis anidados y equilibrados; es decir, cadenas como `(())`, `()()()`, `(())((()))`.

Solución: Es fácil ver que G genera cadenas de paréntesis anidados y equilibrados ya que cada aplicación de la producción $S \to \texttt{(}S\texttt{)}S$ da lugar a un par de paréntesis equilibrados.

Para establecer la dirección recíproca demostraremos por inducción sobre n la siguiente afirmación: “Toda cadena con $2n$ paréntesis anidados y equilibrados se puede generar en G”.

$n = 1$: $\texttt{()}$ se genera con $S \Rightarrow \texttt{(}S\texttt{)}S \overset{2}{\Longrightarrow} \texttt{()}$.

$n = 2$: $\texttt{()()}$ se genera con $S \Rightarrow \texttt{(}S\texttt{)}S \Rightarrow \texttt{(}S\texttt{)(}S\texttt{)}S \overset{3}{\Longrightarrow} \texttt{()()}$.

$\texttt{(())}$ se genera con $S \Rightarrow \texttt{(}S\texttt{)}S \Rightarrow \texttt{((}S\texttt{)}S\texttt{)}S \overset{3}{\Longrightarrow} \texttt{(())}$.

Paso inductivo: una cadena con $2n$ paréntesis anidados y equilibrados es de la forma $\texttt{(}u\texttt{)}$ ó $\texttt{(}u\texttt{)}v$ donde u y v tienen estrictamente menos de $2n$ paréntesis anidados y equilibrados. Por hipótesis de inducción $S \overset{*}{\Longrightarrow} u$ y $S \overset{*}{\Longrightarrow} v$. Por lo tanto,

$$S \Rightarrow \texttt{(}S\texttt{)}S \overset{*}{\Longrightarrow} \texttt{(}u\texttt{)}S \Rightarrow \texttt{(}u\texttt{)}.$$

$$S \Rightarrow \texttt{(}S\texttt{)}S \overset{*}{\Longrightarrow} \texttt{(}u\texttt{)}S \overset{*}{\Longrightarrow} \texttt{(}u\texttt{)}v.$$

Ejercicios de la sección 4.1

1. Encontrar GIC que generen los siguientes lenguajes sobre $\Sigma = \{a, b\}$:

   (i) El lenguaje de las cadenas que tienen un número par de $b$es.

   (ii) El lenguaje de las cadenas que comienzan con b y terminan con ba.

   (iii) $a^*b \cup a$.

   (iv) $a^*b \cup b^*a$.

   (v) $(ab^* \cup b^*a)^*$.

   (vi) $\{a^ib^{2i} : i \geq 0\}$.

   (vii) $\{ab^iab^i : i \geq 1\}$.

2. Encontrar GIC que generen los siguientes lenguajes sobre el alfabeto $\{a, b, c, d\}$:

   (i) $\{a^ib^jc^jd^i : i, j \geq 1\}$.

   (ii) $\{a^ib^ic^jd^j : i, j \geq 1\}$.

   (iii) $\{a^ib^jc^jd^i : i, j \geq 1\} \cup \{a^ib^ic^jd^j : i, j \geq 1\}$.

3. Encontrar una GIC que genere el lenguaje $\{a^ib^jc^{i+j} : i \geq 0,\ j \geq 1\}$, sobre $\{a, b, c\}$.

4. Sea $\Sigma = \{0, 1\}$. Encontrar una GIC que genere el lenguaje de las cadenas que tienen igual número de ceros que de unos.

5. Demostrar que la gramática $S \to SS \mid \texttt{(}S\texttt{)} \mid \lambda$ también genera el lenguaje de todas las cadenas de paréntesis anidados y equilibrados.

6. Encontrar una gramática que genere el lenguaje de todas las cadenas de paréntesis circulares y/o angulares, anidados y equilibrados. Es decir cadenas como `([()])`, `([])[()]`, `[()([[()()]])]`, etc. Cadenas como `([)]` ó `[([)]]` tienen paréntesis equilibrados pero no anidados y, por lo tanto, no pertenecen a este lenguaje.