GRAMÁTICAS.

Una gramática consta de los siguientes elementos: un alfabeto $\mathbf{\Sigma}$, pero ahora le llamaremos conjunto de caracteres **terminales.** Consta también de un conjunto finito N de caracteres llamados **no terminales**. También hay un caracter inicial S. Finalmente un conjunto finito de **producciones.** Toda producción debe tener al menos un caracter no terminal a la izquierda o S.

NOTACIÓN. Los caracteres terminales (también llamado conjunto T) suelen denotarse con minúsculas: a, b,...; los caracteres no terminales con mayúsculas: A, B, C,.... Las producciones son sustituciones de caracteres. Se asemejan a la definición recursiva de los lenguajes, aunque no siempre son recursivos. Algunos ejemplos de producciones son:

$$\begin{aligned} S &\rightarrow \lambda,\ S \rightarrow 01A10,\ S \rightarrow AB, S \rightarrow AC \\ AC &\rightarrow CAB,\ A \rightarrow 1A,\ 1A \rightarrow 11, \\ C &\rightarrow 0,\ B \rightarrow 00. \end{aligned}$$

Todas ellas son con $T = \{0, 1\}$ y $N = \{A, B, C\}$. Con estas producciones podemos generar cadenas en $T^* = \{0, 1\}^*$. Siempre comenzamos con las producciones que tengan S a la izquierda.

Algunas cadenas generadas por esta gramática son $S \rightarrow \lambda$. Es decir, la cadena vacía pertenece al lenguaje generado por esta gramática. Otras cadenas serían

$$\begin{aligned} S &\rightarrow AB \\ &\rightarrow 1AB \\ &\rightarrow 1A00 \\ &\rightarrow 1\mathbf{1A}00 \\ &\rightarrow 1\mathbf{11}00. \end{aligned}$$

Otro ejemplo de cadena sería

$$\begin{aligned} S &\rightarrow AC \\ &\rightarrow CAB \\ &\rightarrow 0AB \\ &\rightarrow 01AB \\ &\rightarrow 01A00 \\ &\rightarrow 01100 \end{aligned}$$

En los ejemplos anteriores, hay caracteres que aparecen a la izquierda de diferentes producciones, eso lo simplificamos como

$$S \rightarrow \lambda|01A10|AB|AC$$

y con esto decimos que son las cuatro producciones $S \rightarrow \lambda,\ S \rightarrow 01A10,\ S \rightarrow AB, S \rightarrow AC$.

EJEMPLO. Con $T = \{0, 1\}$ y $N = \{A, B\}$, desde luego S y las producciones

$$\begin{aligned} S &\rightarrow \lambda|1|0A|1B \\ A &\rightarrow 0A|1B|1 \\ B &\rightarrow 1|1B|0A \end{aligned}$$

Se trata de determinar qué tipo de cadenas se producen. Vemos que para salir de la secuencia de producciones, únicamente con un 1 al final. Comenzamos siempre con S, generamos λ, 1 o continuamos produciendo con $1A$ o $1B$. Por ejemplo

$$\begin{aligned} S &\rightarrow 0A \\ &\rightarrow 00A \\ &\rightarrow 001B \\ &\rightarrow 0010A \\ &\rightarrow 00101 \end{aligned}$$

Notamos que con las producciones $A \rightarrow 0A|1B$ y $B \rightarrow 1B|0A$, antes de finalizar la cadena, podemos generar cualquier cadena binaria para al fina salir con un 1. Entonces esta gramática genera el lenguaje de λ y todas las cadenas binarias que terminan en 1. Estas útlimas son las cadenas con postfijo 1. Su expresión es $\{\lambda\} \cup \{0,1\}^* 1$ o $\{\lambda\} \cup (\mathbf{0},\mathbf{1})^* \mathbf{1}$. Es un ejercicio verificar que la cadena

$$0010010101000101011$$

se puede generar con esta gramática.

EJEMPLO. Con $T = \{0,1,2\}$ y $N = \{A,B,C\}$ proporcione producciones para generar el lenguaje $\{0^n 1^n 2^n | n \geq 0\}$. Importante: este lenguaje no es $\{0\}^* \{1\}^* \{2\}^*$ porque este último tiene cualquier cantidad de los bits. 011122222 está en $\{0\}^* \{1\}^* \{2\}^*$ pero **no** en $\{0^n 1^n 2^n | n \geq 0\}$. Una gramática que genera este lenguaje es como sigue: como son tres bits, parecería que necesitamos tres caracteres no terminales, uno por cada bit.

$$\begin{aligned} S &\rightarrow \lambda|C, \\ C &\rightarrow 0CAB|\lambda \end{aligned}$$

donde la producción $C \rightarrow 0CAB$ genera tantos ceros como deseemos y al mismo tiempo caracteres no terminales A y B en igual número que de ceros. Se intuye que A se sustituirá por 1 y B por 2. Con $C \rightarrow \lambda$ dejamos de producir ceros. Por ejemplo, con $n = 3$

$$\begin{aligned} C &\rightarrow 0\mathbf{C}AB \\ &\rightarrow 0\mathbf{0CAB}AB \\ &\rightarrow 000CABABAB \\ &\rightarrow 000ABABAB. \end{aligned}$$

Ahora debemos convertir A en 1 y B en 2 (o viceversa) pero necestimaos que las A's estén consecutivas y asimismo las B's. Eso se logra con una de las producciones $AB \rightarrow BA$ o $BA \rightarrow AB$. Pero si usamos la primera la B queda enseguida de un cero y debe convertirse en 1. Con la segunda nos quedamos con que A se sustituye por 1 y B por 2. Así, con $BA \rightarrow AB$ al aplicarla sucesivamente obtenemos

$$\begin{aligned} 000A\mathbf{BA}BAB &\rightarrow 000A\mathbf{AB}BAB \\ &\rightarrow 000AABABB \\ &\rightarrow 000AAABBB. \end{aligned}$$

Es muy tentador usar la producciones $A \rightarrow 1$ y $B \rightarrow 2$. Ello no es posible en este lenguaje, pues las producciones se aplican en cualquier orden; así generaríamos la cadena

$$000ABABAB \rightarrow 000121212$$

la cual es no válida, es decir, no pertenece a nuestro lenguaje. Entonces debemos preservar el orden de los bits 1 y 2, lo cual se logra con la producción $0A \to 01$ para generar el primer bit 1.

$$00\mathbf{0A}AABBB \to 00\mathbf{01}AABBB$$

Para el resto usamos $1A \to 11$ cuantas veces sea necesario (pues exactamente el número de A's que fue precisamente el mismo númoero de 0's).

$$\begin{aligned} 000\mathbf{1A}ABBB &\to 000\mathbf{11}ABBB \\ &\to 000111BBB. \end{aligned}$$

La idea se repite par B: el primer 2 lo producimos con $1B \to 12$ y el resto con $2B \to 22$. Continuando con las producciones

$$\begin{aligned} 000111BBB &\to 0001112BB \\ &\to 00011122B \\ &\to 000111222. \end{aligned}$$

Las producciones las reunimos

$$\begin{aligned} S &\to \lambda|C \\ C &\to \lambda|0CAB \\ BA &\to AB \\ 0A &\to 01 \\ 1A &\to 11 \\ 1B &\to 12 \\ 2B &\to 22. \end{aligned}$$

Dado el lenguaje (como dato del ejercicio) obtener las producciones de la gramática que genere el lenguaje es no fácil. Una difucultad es dar más producciones de las necesarias pero que no generen cadenas que no pertenezcan al lenguaje (cadenas no válidas). Otra dificultad es que nos falten producciones y no podamos generar todas las cadenas válidas. Si lo que se da como dato es la gramática, es más sencillo detectar el lenguaje que se genera.

Pasar del autómata finito (AF) al lenguaje es sencillo, pues un AF siempre reconocerá un lenguaje con expresión regular. Los lenguajes regulares tendrán una gramática regular. Se dan los 4 tipos de gramáticas:

TIPO 0 No hay restricciones, excepto por que siempre haya al menos un caracter no terminal en la izquierda.

TIPO 1 Las producciones $\omega_1 \to \omega_2$ son de la forma con $\omega_1 = lAr$ y $\omega_2 = l\omega r$ donde $l, r, \omega \in (N \cup T)^*$ y $\omega \neq \lambda$. Por ejemplo

$$1A00 \to 12B00.$$

Estas gramáticas se llaman **sensibles al contexto**, pues en $1A00 \to 12B00$ la A se cambia por $2B$ cuando y únicamente cuando a la izquierda tiene 1 y a la derecha 00. Además $S \to \lambda$ es permitida pero que haya S la derecha no, es decir, $S \to 0AS$ no es permitida (en TIPO 0 sí). En este tipo S a la derecha se permite siempre que no esté permitida $S \to \lambda$. O una u otra pero no ambas.

TIPO 2 Son las llamadas gramáticas **no sensibles al contexto**, pues a la izquierda únicamente cadenas unitarias de caracteres no terminales. La producción $A \to \omega$ genera $\omega \in (N \cup T)^*$. Se permite $S \to \lambda$ pero nunca S a la derecha.

TIPO 3 Son las gramáticas regulares. Se permite $S \to \lambda$ pero nunca S a la derecha. Las producciones son del tipo

$$\begin{aligned} A &\to 1 \\ A &\to 0B. \end{aligned}$$

EJEMPLOS.

a) Se da la gramática $G = (S, T = \{0,1\}, N = \{A\}, P)$ donde las producciones P son

$$\begin{aligned} S &\to \lambda|0|1A \\ A &\to 0A|1A|1 \end{aligned}$$

La cadena vacía y 0 están en el lenguaje generdado por la gramática, que simbolizamos $\mathcal{L}(G)$. Para más cadenas únicamente empezando con 1, enseguida, las producciones $A \to 0A|1A$ nos permiten generar cualquier cantidad de bits en $\{0,1\}$, es decir, después del bit 1, cualquier cadena en $\{0,1\}^*$ y únicamente podemos terminar la cadena con un 1. Entonces $\mathcal{L}(G)$ son las cadenas λ, 0 y cualquier cadena binaria que comience y termine con 1. Las últimas son cadenas con prefijo propio 1 y postfijo propio 1. La expresión regular es

$$\{\lambda\} \cup (\mathbf{0}) \cup \mathbf{1}\{\mathbf{0},\mathbf{1}\}^*\mathbf{1}.$$

b) Ahora nos dan la expresión regular

$$(\mathbf{0},\mathbf{10})(\mathbf{10})^* \cup (\mathbf{1},\mathbf{01})(\mathbf{01})^*$$

que sabemos son todas las cadenas binarias no vacías sin bits iguales consecutivos. Precisamos de alternar los bits, por ello necesitamos dos caracteres no terminales, A y B. Las producciones son

$$\begin{aligned} S &\to A|B \\ A &\to 1B|1 \\ B &\to 0A|0 \end{aligned}$$

La gramática es $(S, \{0,1\}, \{A,B\}, P)$.

c) La $G = (S, \{0,1\}, P)$ con las producciones $S \to \lambda|0S1$. Es gramática tipo cero. Genera el lenguaje $\mathcal{L}(G)$ de las cadenas:

$$\begin{aligned} S &\to 0S1 \\ &\to 00S11 \\ &\to 000S111 \\ &\to 0000S1111 \\ &\to 00001111 \end{aligned}$$

Notamos que genera todas las cadenas que comienzan con n ceros y terminan con n ceros para $n \geq 0$. Es lenguaje no regular $\{0^n1^n : n \geq 0\}$.

d) Ahora nos dan el lenguaj de cadenas binarias que comienzan con las cadenas 00 o 11 como prefijo propio y con postijos propios también 00 o 11. Su expresión regular es sencilla

$$\{00, 11\}\,\{0, 1\}^*\,\{00, 11\}$$