Autómatas y Lenguajes Formales 2017-1
Facultad de Ciencias UNAM
Nota de Clase 7

Favio E. Miranda Perea A. Liliana Reyes Cabello Lourdes González Huesca

25 de septiembre de 2016

1. Gramáticas

Un mecanismo relevante para generar un lenguaje es mediante el concepto de gramática formal. Las gramáticas formales fueron introducidas por Noam Chomsky en 1956. La intención era tener un modelo para la descripción de lenguajes naturales. Posteriormente se utilizaron como herramienta para presentar la sintaxis de lenguajes de programación y para el diseño de analizadores léxicos de compiladores

Definición 1 *Una gramática es una cuaterna* $G = \langle V, T, S, P\rangle$ *tal que:*

- V *es un alfabeto de* variables *o* símbolos no-terminales*, los cuales se denotan con mayúsculas* $A, B, C, \ldots$
- T *es un alfabeto de* símbolos terminales*, los cuales se denotan con minúsculas* $a, b, c, \ldots$. *Además se requiere que* $T \cap V = \varnothing$.
- $S \in V$ *es una variable distinguida llamada el* símbolo inicial*.*
- P *es un conjunto finito de reglas de reescritura, llamadas* reglas de producción *o producciones.*

El conjunto de reglas de producción P es un conjunto finito de pares $\langle \alpha, \beta\rangle$ tales que

- $\alpha \in (V \cup T)^\star - T^\star$. Es decir, α es una cadena de símbolos terminales o no terminales, con al menos un símbolo no-terminal.
- $\beta \in (V \cup T)^\star$. Es decir, β es una cadena de símbolos de $V \cup T$, los cuales podrán ser todos terminales.

En lugar de escribir $\langle \alpha, \beta\rangle \in P$, escribimos

$$\alpha \to \beta$$

y decimos que α **produce** a β, o que α se **reescribe** en β.

Las reglas de producción sirven para generar cadenas, proceso que se formaliza mediante las derivaciones formales:

Definición 2 (Derivación formal) *Dadas dos palabras $w, v \in (V \cup T)^\star$ decimos que v es* derivable *a partir de w en* ***un paso*** *($w \to v$) si y sólo si:*

> *Existe una regla $\alpha \to \beta$ en P y cadenas $\gamma_1, \gamma_2 \in (V \cup T)^\star$ tales que:*
>
> $$w = \gamma_1 \alpha \gamma_2 \qquad\qquad v = \gamma_1 \beta \gamma_2$$
>
> *Es decir, se ha reescrito la cadena α en la cadena β.*

Algunos autores utilizan $\Rightarrow$ en vez de $\to$ para denotar la relación de derivación. Nosotros preferimos sobrecargar el operador $\to$.

Definición 3 *Decimos que una cadena v es* derivable *a partir de w si existen palabras $\gamma_2, \ldots, \gamma_n$ tales que*

$$w = \gamma_1 \to \gamma_2 \ldots \gamma_{n+1} \to \gamma_n = v$$

En tal caso escribimos $w \to^\star v$.

Definición 4 (Lenguaje de una gramática) *Dada una gramática $G = \langle V, T, S, P \rangle$ definimos al lenguaje generado por G, denotado $L(G)$, como el conjunto de palabras de símbolos* ***terminales*** *derivables a partir del símbolo inicial S. Es decir,*

$$L(G) = \{w \in T^\star \mid S \to^\star w\}$$

Ejemplo: Considere el siguiente lenguaje $L = (a+b)^\star$, es el lenguaje en que cualquier cadena con a y b debe generarse. Veamos la gramática correspondiente:

- La cadena vacía debe generarse: $S \to \varepsilon$
- Si $w \in L$ entonces $wa \in L$: $S \to Sa$
- Si $w \in L$ entonces $wb \in L$: $S \to Sb$

La cadena $w = ababb$ se deriva como sigue:

$$S \to Sb \to Sbb \to Sabb \to Sbabb \to Sababb \to ababb$$

Ejemplo: Proporcionar una gramática que genere las cadenas en $L = \{a^i b^j \mid i, j \in \mathbb{N}, i \leq j\}$.

- La cadena vacía debe generarse ($i = j = 0$): $S \to \varepsilon$
- Debe haber al menos tantas b como a, primero las a y luego las b:

  $$S \to aSb$$

- Puede haber más b al final:

  $$S \to Sb$$

Ejemplo de derivación para la cadena $w = aabbb$

$$S \to aSb \to aaSbb \to aaSbbb \to aabbb$$

Ejemplo: El lenguaje $L = \{a^i b^j a^j b^i \mid i, j \in \mathbb{N}\}$ es generado por la siguiente gramática:

- Primero generamos el centro de la palabra, $b^j a^j$:

$$S \to B \qquad B \to bBa \qquad B \to \varepsilon$$

- Después los extremos a^i, b^i: $\qquad S \to aSb$

Un ejemplo de derivación es la generación de la cadena $w = aababb$

$$S \to aSb \to aaSbb \to aaBbb \to aabBabb \to aababb$$

Ejemplo: Dado $L = \{a^i b^i a^j b^j \mid i, j \in \mathbb{N}\}$, la siguiente gramática genera el lenguaje:

- Primero generaremos el lenguaje $\{a^i b^i \mid i \in \mathbb{N}\}$ mediante:

$$P \to \varepsilon \qquad\qquad P \to aPb$$

- Para generar a L simplemente agregamos:

$$S \to PP$$

Ejemplo de derivación, la cadena $w = aabbab$ se genera como sigue:

$$S \to PP \to aPbP \to aPbaPb \to aaPbbaPb \to aaPbbab \to aabbab$$

Ejemplo: El lenguaje $L = \{a^i b^i \mid i \in \mathbb{N}\} \cup \{b^i a^i \mid i \in \mathbb{N}\}$ se genera mediante:

- Primero el lenguaje $\{a^i b^i \mid i \in \mathbb{N}\}$ se genera con la siguiente gramática:

$$P \to \varepsilon \qquad\qquad P \to aPb$$

- Después generamos el lenguaje $\{b^i a^i \mid i \in \mathbb{N}\}$ mediante:

$$Q \to \varepsilon \qquad\qquad Q \to bQa$$

- Finalmente, para generar a L simplemente agregamos:

$$S \to P \qquad S \to Q$$

Ejemplo de derivación para $w = bbbaaa$

$$S \to P \to aPb \to aaPbb \to aaaPbbb \to aaabbb$$

1.1. Diseño de gramáticas

Si bien muchas veces el diseño de una gramática G para un lenguaje dado L es intuitivamente claro y correcto, esto debe mostrarse formalmente, mostrando que $L = L(G)$. Esto se hace, por supuesto, probando formalmente lo siguiente:

1. **Correctud**: la gramática G genera únicamente las cadenas de L, es decir, $L(G) \subseteq L$.
2. **Completud**: toda cadena de L es generada por G, es decir, $L \subseteq L(G)$.

2. Jerarquía de Chomsky

Las gramáticas fueron clasificadas de acuerdo a sus propiedades por Chomsky.

Tipo 0 **Lenguajes recursivamente enumerables**
Son aquellos lenguajes generados por una gramática sin restricciones adicionales.
Tales gramáticas pueden incluir reglas de la forma $\boxed{\alpha \to \varepsilon}$

De manera que la gramática es capaz de borrar cadenas. Tales gramáticas se conocen como *contraíbles*. Por ejemplo:

$$aS \to bSb,\ aSb \to \varepsilon,\ SbS \to bcS$$

Así también la siguiente es una gramática de tipo 0 donde $L(G) = \{ww \mid w \in \{0,1\}^\star\}$

$$\begin{array}{cccc} S \to AT & A \to 0AO & A \to 1AI & O0 \to 0O \\ O1 \to 1O & I0 \to 0I & I1 \to 1I & OT \to 0T \\ IT \to 1T & A \to \varepsilon & T \to \varepsilon & \end{array}$$

La idea del diseño de esta gramática y la razón del nombre *recursivamente enumerable* se discutirán más adelante.

Comentamos ahora que las máquinas que aceptan este tipo de lenguajes son las Máquinas de Turing que también serán estudiadas más adelante.

Tipo 1 **Lenguajes dependientes del contexto**
También llamados sensibles al contexto, son aquellos lenguajes generados por gramáticas con todas sus producciones son de la forma

$$\boxed{\alpha_1 A \alpha_2 \to \alpha_1 \beta \alpha_2}$$

con $\alpha_1, \alpha_2 \in (V \cup T)^\star,\ A \in V,\ \beta \neq \varepsilon$, además de que $|\alpha_1 \beta \alpha_2| \geq |\alpha_1 A \alpha_2|$.
Con la posible excepción de la regla $S \to \varepsilon$, en cuyo caso se prohibe la presencia de S a la derecha de las producciones.

Los autómatas que reconocen este tipo de lenguajes son los llamados autómatas acotados linealmente.

Veamos un ejemplo, la siguiente gramática dependiente del contexto genera al lenguaje

$$L = \{a^i b^i c^i \mid i \geq 0\}$$

$$\begin{array}{rcl@{\qquad}rcl} S & \to & A & A & \to & aABC \mid abC \\ CB & \to & BC & bB & \to & bb \\ bC & \to & bc & cC & \to & cc \end{array}$$

Tipo 2 **Lenguajes libres del contexto**

Son aquellos generados por gramáticas con todas sus producciones de la forma

$$\boxed{A \to \alpha}$$

con $A \in V,\ \alpha \in (V \cup T)^{\star}$.

Esta definición incluye a la regla $S \to \boldsymbol{\varepsilon}$. La mayoría de las gramáticas para lenguajes de programación caen en esta categoría. Así mismo las máquinas que aceptan este tipo de lenguajes son los llamados autómatas de pila que serán estudiados en el siguiente tema.

Tipo 3 **Lenguajes regulares** Son aquellos generados por una gramática de una de las siguientes formas:

- Lineal por la derecha: todas las producciones de la forma

  $$A \to aB \quad A \to a \quad A \to \boldsymbol{\varepsilon}$$

  con $A, B \in V,\ a \in T$

- Lineal por la izquierda: todas las producciones de la forma

  $$A \to Ba \quad A \to a \quad A \to \boldsymbol{\varepsilon}$$

  con $A, B \in V,\ a \in T$

- **No** se permite mezclar ambos tipos de producciones.

Estos lenguajes son los que hemos estudiado a detalle hasta ahora y como hemos visto las máquinas que los aceptan son los autómatas finitos.

2.1. Observaciones

Decimos que un lenguaje es de tipo i si y sólo si i es el índice *más grande* tal que existe una gramática de tipo i que genera a L. Es decir que si un lenguaje L es generado por una gramática de tipo i, no se puede asegurar de inmediato que L sea un lenguaje de tipo i. Debe asegurarse que i es *máximo*. Para ello se puede ver la jerarquía generada por la clasificación de Chomsky:

$$\mathcal{L}_3 \subsetneq \mathcal{L}_2 \subsetneq \mathcal{L}_1 \subsetneq \mathcal{L}_0$$

La jerarquía de Chomsky permite refinar la teoría de la computación clasificando lenguajes en función de los recursos computacionales necesarios para reconocerlos. Como vimos cada categoría tiene asociado un tipo de máquina que reconoce a los lenguajes.