4.3. Gramáticas ambiguas

La noción de ambigüedad se puede presentar en términos de árboles sintácticos o en términos de ciertas derivaciones "estándares": las llamadas derivaciones a izquierda.

4.3.1 Definición. Una derivación se llama **derivación a izquierda** (o derivación más a la izquierda) si en cada paso se aplica una producción a la variable que está más a la izquierda.

Una derivación cualquiera se puede transformar siempre en una única derivación a izquierda aplicando, en cada paso, producciones a la variable que esté más a la izquierda.

4.3.2 Definición. Una GIC G es **ambigua** si existe una cadena $w \in \Sigma^*$ para la cual hay dos derivaciones a izquierda diferentes. Equivalentemente, una GIC G es **ambigua** si existe una cadena $w \in \Sigma^*$ con dos árboles de derivación diferentes.

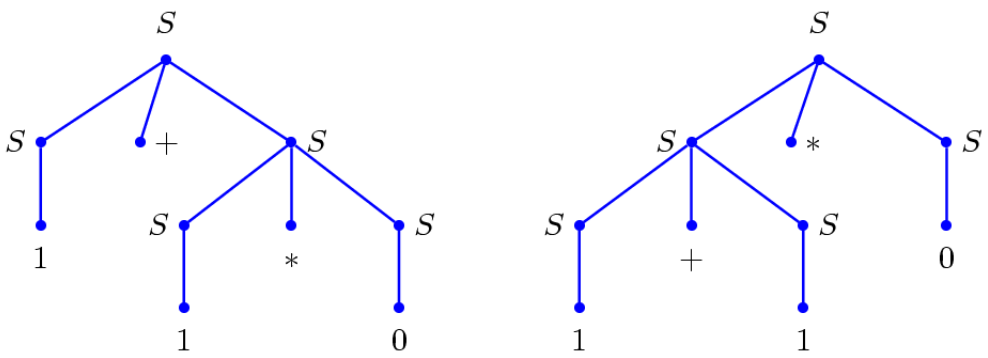
Ejemplo Considérese el alfabeto $\Sigma = \{0, 1, +, *, (,)\}$. La siguiente gramática G_{sp} para generar números naturales, sumas y productos, en numeración binaria, es ambigua:

$$S \to S + S \mid S * S \mid (S) \mid 0S \mid 1S \mid 0 \mid 1$$

La cadena $1 + 1 * 0$ tiene dos derivaciones a izquierda diferentes:

$$S \Rightarrow S + S \Rightarrow 1 + S \Rightarrow 1 + S * S \Rightarrow 1 + 1 * S \Rightarrow 1 + 1 * 0.$$
$$S \Rightarrow S * S \Rightarrow S + S * S \Rightarrow 1 + S * S \Rightarrow 1 + 1 * S \Rightarrow 1 + 1 * 0.$$

Los árboles de derivación correspondientes a las anteriores derivaciones son:



En la gramática G_{sp} la ambigüedad se puede eliminar introduciendo paréntesis:

$$S \to (S + S) \mid (S * S) \mid (S) \mid 0S \mid 1S \mid 0 \mid 1$$

Aunque la introducción de paréntesis elimina la ambigüedad, las expresiones generadas tienen un excesivo número de paréntesis lo que dificulta el análisis sintáctico (en un compilador, por ejemplo). Lo más corriente en estos casos es utilizar gramáticas ambiguas como G_{sp} siempre y cuando se establezca un orden de precedencia para los operadores. Lo usual es establecer que $*$ tenga una mayor orden de precedencia que $+$, es decir, por convención $*$ actúa antes que $+$. Por ejemplo, la expresión $1*1+0$ se interpreta como $(1*1)+0$ y la expresión $10+11*110+1$ se interpreta como $10+(11*110)+1$.

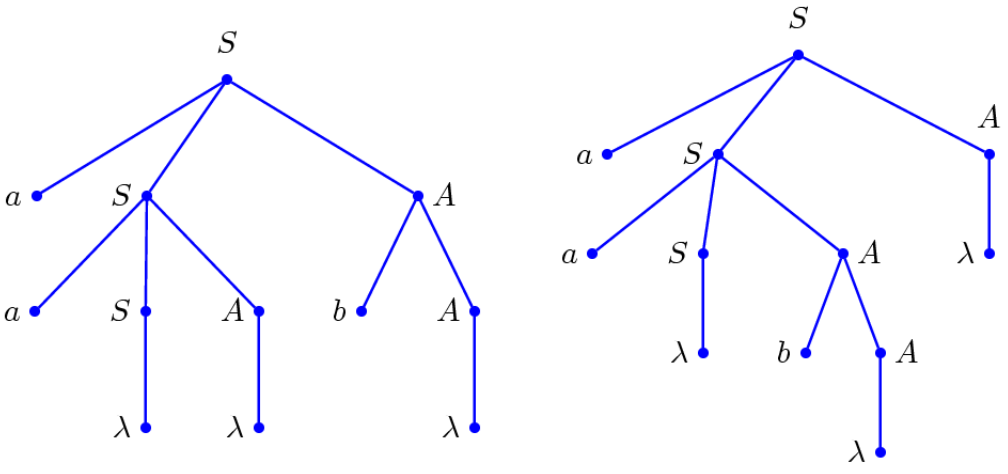
Ejemplo La siguiente gramática es ambigua:

$$G: \quad \begin{cases} S \to aSA \mid \lambda \\ A \to bA \mid \lambda \end{cases}$$

G es ambigua porque para la cadena aab hay dos derivaciones a izquierda diferentes.

$$S \Rightarrow aSA \Rightarrow aaSAA \Rightarrow aaAA \Rightarrow aaA \Rightarrow aabA \Rightarrow aab.$$
$$S \Rightarrow aSA \Rightarrow aaSAA \Rightarrow aaAA \Rightarrow aabAA \Rightarrow aabA \Rightarrow aab.$$

Los árboles de derivación para estas dos derivaciones son:



El lenguaje generado por esta gramática es $a^+b^* \cup \lambda$. Se puede construir una gramática no-ambigua que genere el mismo lenguaje:

$$G' : \quad \begin{cases} S \to AB \mid \lambda \\ A \to aA \mid a \\ B \to bB \mid \lambda \end{cases}$$

Para ver que la gramática G' no es ambigua se puede razonar de la siguiente manera: la cadena λ se puede generar de manera única con la derivación $S \Rightarrow \lambda$. Una derivación de una cadena no vacía debe comenzar aplicando la producción $S \to AB$; la variable A genera cadenas de $a$es de manera única y B genera cadenas de $b$es también de manera única. Por consiguiente, toda cadena tiene una única derivación a izquierda.

✎ Existen lenguajes independientes del contexto para los cuales toda gramática es ambigua. Tales lenguajes se llaman **inherentemente ambiguos**. Un ejemplo es el lenguaje

$$L = \{a^i b^i c^j d^j : i, j \geq 1\} \cup \{a^i b^j c^j d^i : i, j \geq 1\}.$$

sobre el alfabeto $\{a, b, c, d\}$. En el Ejercicio 4.1.2. (iii) se pidió mostrar que L es un LIC. La demostración de que L es inherentemente ambiguo es bastante intrincada y puede encontrarse en la referencia [HU].

✎ No existe ningún algoritmo que permita determinar si una GIC dada es o no ambigua. Con la terminología de la sección 3.6, esto quiere decir que el problema de la ambigüedad para GIC es indecidible. Éste no es un resultado trivial; para su demostración remitimos al lector a la referencia [HMU].

Ejercicios de la sección 4.3

1. Mostrar que las siguientes gramáticas son ambiguas:

   (i) $S \to aSbS \mid bSaS \mid \lambda$.

   (ii) $S \to abS \mid abScS \mid \lambda$.

2. Mostrar que las gramáticas G_1 y G_2 siguientes son ambiguas. En cada caso, encontrar el lenguaje generado por la gramática y construir luego una

gramática no-ambigua que genere el mismo lenguaje.

$$G_1: \quad \begin{cases} S \to AaSbB \mid \lambda \\ A \to aA \mid a \\ B \to bB \mid \lambda \end{cases}$$

$$G_2: \quad \begin{cases} S \to ASB \mid AB \\ A \to aA \mid a \\ B \to bB \mid \lambda \end{cases}$$

3. Encontrar una GIC no-ambigua que genere el lenguaje $a^*b(a \cup b)^*$.