Una **gramática** $G = (V, T, P, S)$ se dice **independiente de contexto o de tipo 2** si y solo si todas las producciones son de la forma:

$$A \rightarrow \alpha$$

donde $A \in V$ y $\alpha \in (V \cup T)^*$. Generan los denominados lenguajes independientes de contexto.

En este tema, vamos a usar un **árbol de derivación**, que es un conjunto de nodos con símbolos del alfabeto, donde se efectúa una ramificación por cada producción que se aplique.

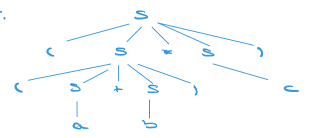
Supongamos la siguiente gramática independiente del contexto:

$S \rightarrow a | b | c$

$S \rightarrow (S+S)$

$S \rightarrow (S*S)$

- Derivación: $S \Rightarrow (S*S) \Rightarrow ((S+S)*S) \Rightarrow ((a+S)*S) \Rightarrow ((a+b)*S)) \Rightarrow ((a+b)*c)$
- Árbol:



### Construcción Árbol de Derivación

Cada nodo del árbol va a contener un símbolo, siendo la raíz el símbolo inicial $S$. Se efectúa una ramificación por cada producción que se aplique. Si a la variable de un nodo, $A$, se le aplica una determinada regla $A \rightarrow \alpha$, entonces para cada símbolo que aparezca en $\alpha$ se añade un hijo con el símbolo correspondiente, situados en el orden de izquierda a derecha. Si la parte derecha es una cadena vacía, entonces se añade un solo hijo, etiquetado $\varepsilon$.

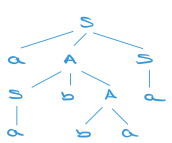
En cada momento, leyendo los nodos de izquierda a derecha se lee la palabra generada

Se llama **derivación por la izquierda** asociada a un árbol a aquella en la que siempre se deriva primero la primera variable (más a la izquierda) que aparece en la palabra.

Análogamente, se llama **derivación por la derecha** asociada a un árbol a aquella en la que se deriva siempre la última variable (más a la derecha) que aparece en la palabra.

Dado la gramática $S \rightarrow aAS$, $S \rightarrow a$, $A \rightarrow SbA$, $A \rightarrow SS$, $A \rightarrow ba$:

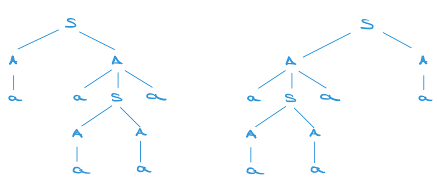
Árbol de derivación



- Derivación por la izquierda: $S \Rightarrow aAS \Rightarrow aSbAS \Rightarrow aabAS \Rightarrow aabbaS \Rightarrow aabbaa$
- Derivación por la derecha: $S \Rightarrow aAS \Rightarrow aAa \Rightarrow aSbAa \Rightarrow aSbbaa \Rightarrow aabbaa$

Una **gramática** se dice **ambigua** si existe una palabra con dos árboles de derivación distintos.

Consideremos la gramática $S \rightarrow AA$, $A \rightarrow aSa$, $A \rightarrow a$ es ambigua, ya que la palabra $a^5$ tiene los dos siguientes árboles de derivación:



Un **lenguaje** de tipo 2 se dice **inherentemente ambiguo** si toda gramática que lo genera es ambigua.

① Determinar si la siguiente gramática es ambigua y si el lenguaje generado es inherentemente ambiguo:

$$S \to A_1 \mid A_2$$
$$A_1 \to aA_1b \mid aA_1 \mid \varepsilon$$
$$A_2 \to aA_2b \mid A_2b \mid \varepsilon$$

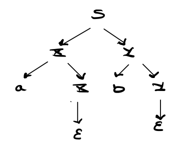
Para ver que la gramática es ambigua tenemos que encontrar una palabra con dos árboles de derivación distintos. En este caso, tomando la palabra ab:

Luego, como hemos encontrado dos árboles de derivación para la palabra ab $\Rightarrow$ G es ambigua.

Observemos que el lenguaje generado por la gramática es $L(G) = \{a^i b^j : i,j \geq 0\}$. Sabemos que $L(G)$ es inherentemente ambiguo $\Leftrightarrow$ Todas las gramáticas que generan dicho lenguaje son ambiguas.

Consideremos la gramática:

$$S \to XY$$
$$X \to aX \mid \varepsilon$$
$$Y \to bY \mid \varepsilon$$



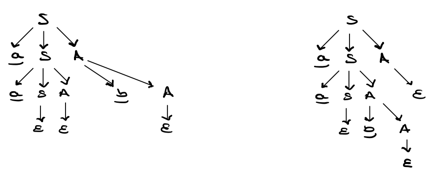
No es ambigua $\Rightarrow$ $L(G)$ no es inherentemente ambiguo

② Sea la gramática

$$S \to aSA \mid \varepsilon \quad , \quad A \to bA \mid \varepsilon$$

a) Demostrar que es ambigua.

Para demostrar que es ambigua buscamos una palabra que tenga asociada dos árboles de derivación distintos. Por ejemplo, la palabra aab:



Luego, como hemos encontrado una palabra, generada por la gramática dada, con dos árboles de derivación distintos $\Rightarrow$ la gramática es ambigua.

b) Dar una expresión regular para el lenguaje generado.

$L(G) = \{a^i b^j : i,j \geq 0\} \Rightarrow L \equiv a^* \cancel{b^*}$   ¡Cuidado!   $a^* + a^+ b^*$

c) Construir una gramática no ambigua, que genere el mismo lenguaje

Gramática idéntica al ejercicio anterior.

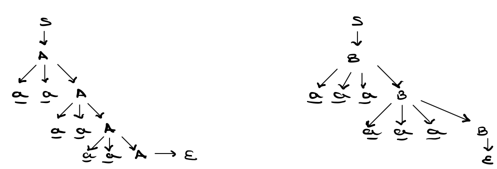
$$S \to aA \mid \varepsilon$$
$$A \to aA \mid bB \mid \varepsilon$$
$$B \to bB \mid \varepsilon$$

④ Dada la gramática

$$S \to A \mid B$$
$$A \to aaA \mid \varepsilon$$
$$B \to aaaB \mid \varepsilon$$

a) Demostrar que es ambigua.

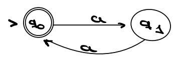
Nuevamente, para demostrar que es ambigua, buscamos una palabra, generada por la gramática, que acepte dos árboles de derivación distintos. Por ejemplo, la palabra $aaaaaa \equiv a^6$ :



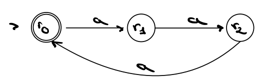
Luego, la gramática es ambigua.

b) Construir un autómata finito determinista que acepte el mismo lenguaje:

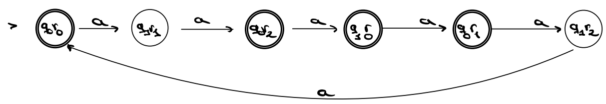
La producción A genera palabras con número de a's par:



La producción B genera palabras con número de a's múltiplo de 3:

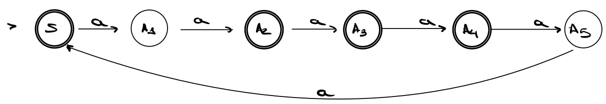


Luego, el autómata resultante (autómata producto)



c) Construir una gramática lineal por la derecha, a partir del autómata determinista anterior, que genere el mismo lenguaje.

Renombrando los estados del autómata:



$S \to aA_1 \mid \varepsilon$

$A_1 \to aA_2$

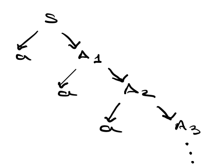
$A_2 \to aA_3 \mid \varepsilon$

$A_3 \to aA_4 \mid \varepsilon$

$A_4 \to aA_5 \mid \varepsilon$

$A_5 \to aS$

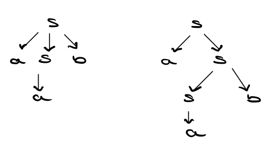
Como observamos la gramática es lineal por la derecha y como cada símbolo solo lleva a una producción no nula, no podrá ser ambiguo, ya que cualquier árbol de derivación es:



⑥ Determinar cuales de las siguientes gramáticas son ambiguas y, en su caso, comprobar si los lenguajes generados son inherentemente ambiguos:

a) $S \to aSb \mid Sb \mid aS \mid a$

Por ejemplo, la palabra aab:



Como hemos encontrado dos árboles de derivación, asociados a una palabra generada por la gramática, distintos ⇒ La gramática es ambigua.

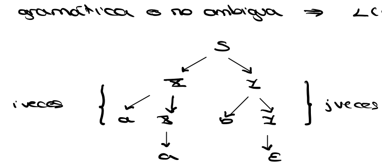
El lenguaje generado por la gramática anterior es $L(G) = \{a^i b^j : i > 0, j \geq 0\} \equiv a^+ b^*$, luego, podemos considerar la siguiente gramática:

$S \to ZY$

$Z \to aZ \mid a$

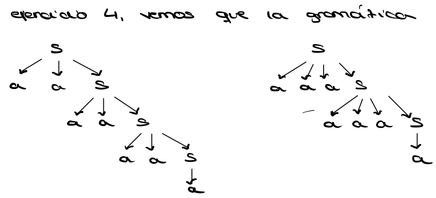
$Y \to bY \mid \varepsilon$

Esta gramática no es ambigua ⇒ $L(G)$ no es inherentemente ambiguo.




b) $S \to aaS \mid aaaS \mid a$

Tomando el mismo caso que en el ejercicio 4, vemos que la gramática es ambigua:

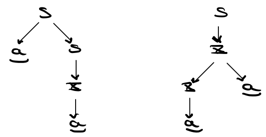


No porque $L = (aa + aaa)^* a$ , por lo que podemos encontrar una gramática regular no ambigua $\Rightarrow$ No es lenguaje inherentemente ambiguo.

c) $S \to aS \mid aSb \mid X$

$X \to Xa \mid a$

Tomando la palabra aa, vemos que la gramática es ambigua, ya que:

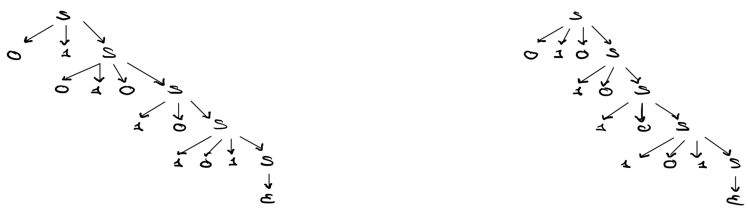


⑦ Dada la gramática

$$S \to 01S \mid 010S \mid 101S \mid \varepsilon$$

determinar si es ambigua. Construir un autómata finito determinista asociado y calcular la gramática lineal por la derecha que se obtiene a partir del autómata. ¿Es ambigua la gramática resultante?

Comenzamos observando que la palabra 01010 10101 tiene dos árboles de derivación asociados:



Luego, la gramática es ambigua.

Pendiente

⑨ Demostrar que la gramática $S \to A1B$, $A \to 0A \mid \varepsilon$, $B \to 0B \mid 1B \mid \varepsilon$ no es ambigua. Encontrar una gramática para el mismo lenguaje que sea ambigua y demostrar su ambigüedad.

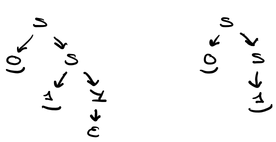
**Nota**: si un lenguaje es regular $\Rightarrow$ Acepta un autómata finito determinista $\Rightarrow$ Sólo se puede derivar siguiendo el autómata $\Rightarrow$
$\Rightarrow$ Gramática no ambigua

$S \to A1B$
$A \to 0A \mid \varepsilon$
$B \to 0B \mid 1B \mid \varepsilon$

$L = 0^* 1 (0+1)^*$ $\Rightarrow$ Como es regular, no es ambigua.

Gramática ambigua:

$S \to 0S \mid 1X \mid 1$
$X \to 0X \mid 1X \mid \varepsilon$



Forma Normal de Chomsky

Cuando las producciones son de la forma

$$A \to BC \;, \; A \to a \quad , \quad A, B, C \in V, \; a \in T$$

1) Aplicar algoritmo para eliminar producciones nulas

Almacenamos en un conjunto las variables que llevan al vacío y sustituye en las variables todas las posibilidades (S es anulable si genera $\varepsilon$)

2) Algoritmo para eliminar producciones unitarias

Creamos un conjunto con $\{(A,S)\}$, que indica que existe la producción unitaria $A \to S$. Una vez creado, A debe poder ir a lo que vaya S.

3) Algoritmo para obtener Chomsky a partir de una gramática sin producciones nulas ni unitarias

- Eliminar los terminales unitarios creando una variable para cada uno.
- Producciones con más de dos símbolos separados: $A \to BEFG \;\longrightarrow\; A \to BD_1 \;,\; D_1 \to EFG$ y eliminamos BEFG

4) Algoritmo para obtener la forma normal de Greibach

Forma $A \to a\alpha, \quad a \in T, \ \alpha \in V^*$

$A \to \alpha, \quad a \in V^*, \ |\alpha| \geq 2$

Ejemplo Youtube

En la forma normal de Chomsky tenemos $\left\} \begin{array}{l} A \to BC \\ A \to a \end{array} \right.$, $A, B, C \in V, \ a \in T$

1) Eliminar $\lambda$- productions y unit- productions

$S \to aA \mid B \mid \lambda$

$\cancel{B \to H}$ Como no lleva a nada, la eliminamos

$A \to a$

2) Agregar nuevas reglas de producción para separar variables y símbolos terminales

| | | | |
|---|---|---|---|
| $S \to ABa$ | | $S \to ABW$ | $W \to a$ |
| $A \to aab$ | $\Rightarrow$ | $A \to WWX$ | $X \to b$ |
| $B \to Ac$ | | $B \to AZ$ | $Z \to c$ |

3) Agregar nuevas variables para quitar variables extras en producciones

| | | | | | |
|---|---|---|---|---|---|
| $S \to A\underline{BW}$ | $W \to a$ | | $S \to AB_1$ | | |
| $A \to W\underline{WX}$ | $X \to b$ | $\Rightarrow$ | $B_1 \to BW$ | $W \to a$ | |
| $B \to AZ$ | $Z \to c$ | | $A \to WW_1$ | $X \to b$ | FNC |
| | | | $W_1 \to WX$ | $Z \to c$ | |
| | | | $B \to AZ$ | | |

Puedes colocar tantas variables como quieras

Ahora, la forma normal de Greibach tiene producciones de la forma $A \to a\alpha$, donde $\boxed{\alpha \in V^*}$ y $a \in T$

1) Hacer, consecutivamente, asignación de variables.

$\left. \begin{array}{l} S \to aS\underbrace{bb}_{A} \\ S \to a \end{array} \right\} \Rightarrow \left. \begin{array}{l} S \to aSA \\ A \to bb \\ S \to a \end{array} \right\} \Rightarrow \begin{array}{l} S \to aSA \mid a \\ A \to bB \\ B \to b \end{array}$

12 Pasar a forma normal de Greibach la gramática

$S \to AAA, \quad S \to B$

$A \to aA, \quad A \to B$

$B \to \varepsilon$