

Árboles de Derivación (parse trees)

Alan Reyes-Figueroa

Teoría de la Computación

(Aula 13) 01.septiembre.2025

Definiciones

Relación entre derivaciones

leftmost y *rightmost*

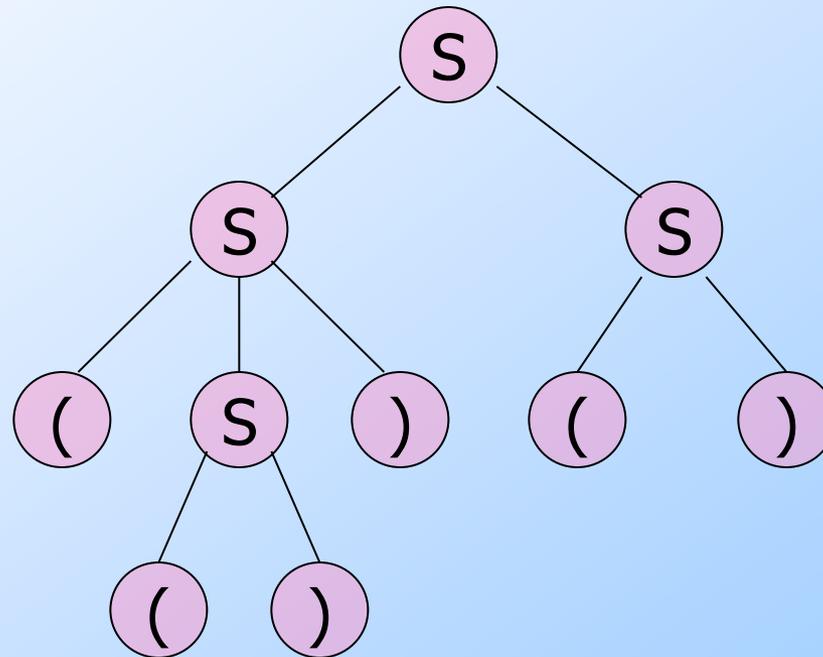
Ambigüedad en Gramáticas

Árboles de Derivación

- *Parse trees* : son árboles etiquetados por los símbolos de una CFG.
- **Hojas**: (nodos terminales), son etiquetados por un terminal o por ϵ .
- **Nodos interiores**: son etiquetados por una variables.
 - Los nodos hijos son etiquetados por por el lado derecho de una regla de producción.
- **Nodo raíz**: etiquetado por el símbolo inicial.

Ejemplo: Parse Tree

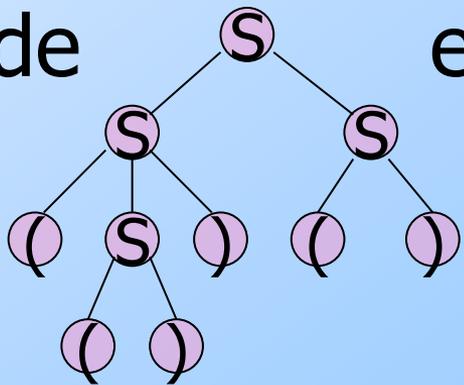
$S \rightarrow SS \mid (S) \mid ()$

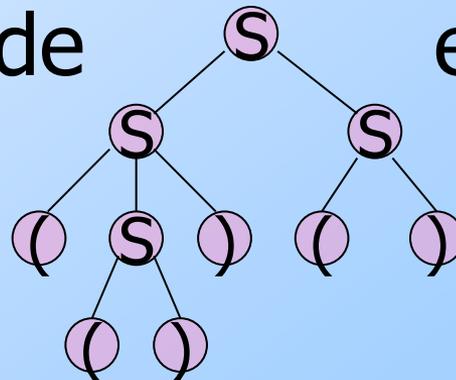


Producción de un Parse Tree

- Concatenación de las etiquetas de las hojas, en orden de izquierda a derecha
 - Esto es, en el orden de un pre-orden transversal.

se llama la *producción* de un parse tree.

- **Ejemplo:** la prod. de  es $((()))()$



Árboles y derivaciones *leftmost*

- **Propiedad:** Para cada parse tree, existe una única derivación a la izquierda, y una única derivación a la derecha, que lo produce.
- **Mostraremos:**
 1. Si hay un parse tree con raíz etiquetada por A y producción w , entonces $A \Rightarrow_{lm}^* w$.
 2. Si $A \Rightarrow_{lm}^* w$, entonces hay un parse tree con raíz A y producción w .

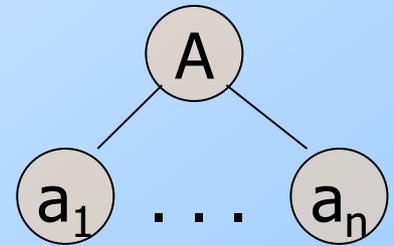
Prueba – Parte 1

□ Por inducción sobre la *altura* del árbol (longitud del mayor trayecto a partir de la raíz).

□ **Base:** Altura 1. El árbol se ve

□ $A \rightarrow a_1 \dots a_n$ es su producción.

□ Luego, $A \Rightarrow^*_{lm} a_1 \dots a_n$.

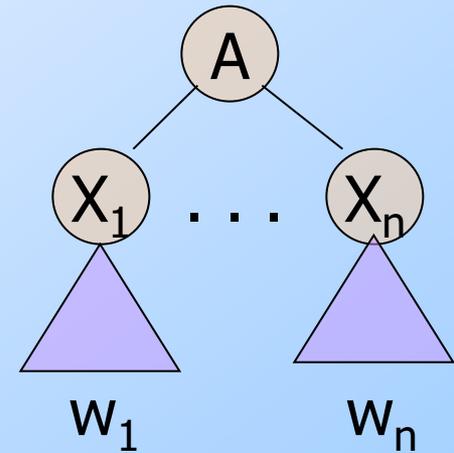


Prueba – Inducción

□ Asuma (1) para árboles del altura $< h$, y suponga que T es de altura h :

□ Por HI, $X_i \Rightarrow^*_{lm} W_i$.

□ Nota: si X_i es terminal, entonces $X_i = W_i$.



□ Así, $A \Rightarrow_{lm} X_1 \dots X_n$
 $\Rightarrow^*_{lm} W_1 X_2 \dots X_n$
 $\Rightarrow^*_{lm} W_1 W_2 X_3 \dots X_n$
 $\Rightarrow^*_{lm} \dots \Rightarrow^*_{lm} W_1 \dots W_n$.

Prueba: Parte 2

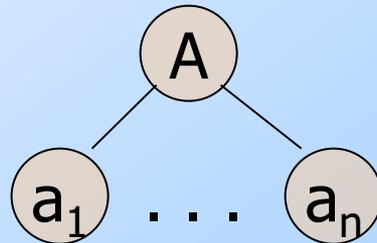
- Dada la derivación *leftmost* de una cadena terminal, ahora debemos mostrar la existencia de un árbol sintáctico que produce dicha cadena.
- La prueba, de nuevo, es por inducción, ahora sobre la longitud de la derivación.

Parte 2 – Base

- Si $A \Rightarrow_{lm}^* a_1 \dots a_n$ es una derivación en un solo paso, esto es

$$A \Rightarrow_{lm} a_1 \dots a_n$$

entonces debemos tener un árbol de derivación

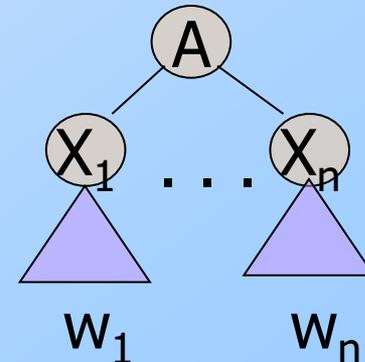


Parte 2 – Inducción

- Asuma (2) para derivaciones que consisten de menos de $k > 1$ pasos, y sea $A \Rightarrow_{Im}^* w$ una derivación en k -pasos.
- El primer paso es $A \Rightarrow_{Im} X_1 \dots X_n$.
- **Punto clave:** w puede dividirse de forma que la primera parte es derivada de X_1 ; la siguiente, derivada de X_2 ; etc.
 - Si X_i es terminal, entonces $w_i = X_i$.

Inducción – (2)

- Esto es, $X_i \Rightarrow^*_{lm} w_i$ para cada i tal que X_i es una variable.
- Además, cada derivación $X_i \Rightarrow^*_{lm} w_i$ toma a lo sumo k pasos.
- Por la HI, si X_i es una variable, existe un árbol con raíz X_i y producción w_i .
- Portanto, tenemos un árbol



Árboles y derivaciones *rightmost*

- Para la prueba de la existencia de derivaciones *rightmost*, las ideas son esencialmente las imagen especular de la prueba para derivaciones *leftmost*.
- Se deja como ejercicio!

Árboles y derivaciones

□ Nota:

La prueba de que se puede obtener un árbol sintáctico a partir de una derivación *leftmost*, realmente no depende de que sea "*leftmost*".

□ Primer paso: debe ser $A \Rightarrow X_1 \dots X_n$.

□ w todavía puede dividirse en porciones, con la primera derivada de X_1 , la siguiente de X_2 , y demás.

Gramáticas Ambiguas

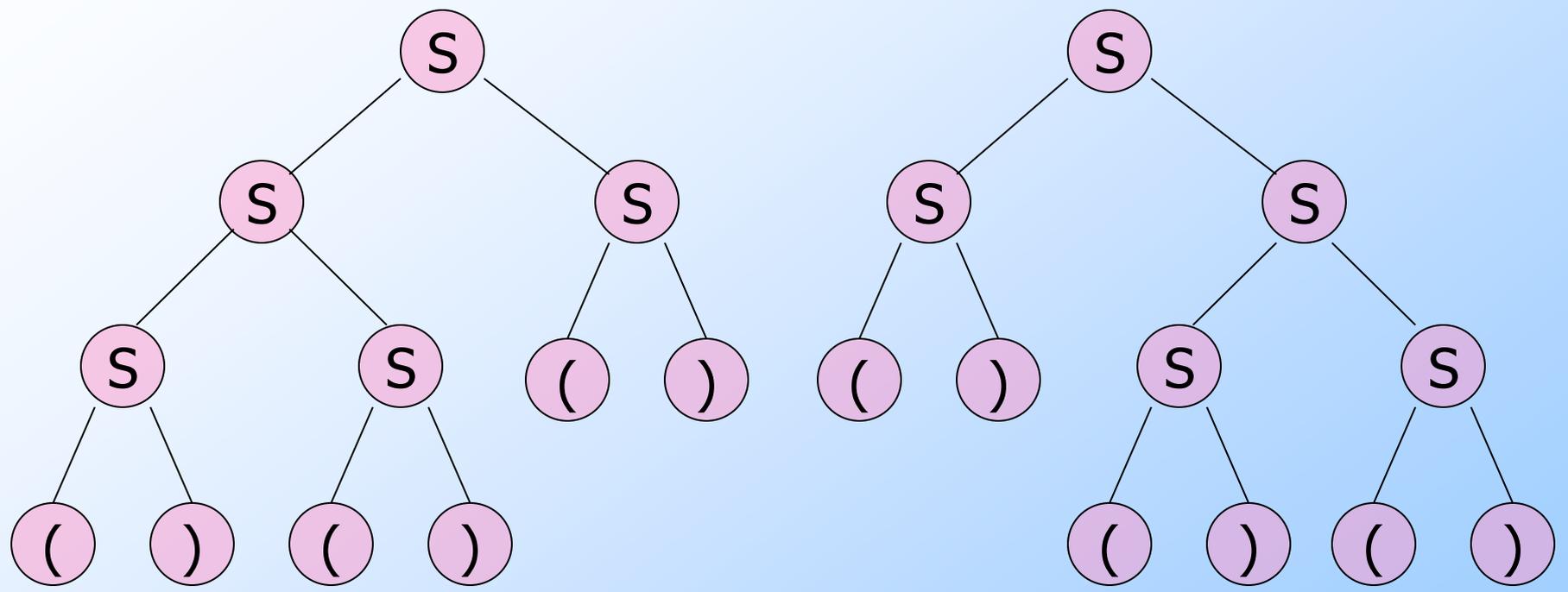
□ Una CFG G es *ambigua* si existe una cadena w en el lenguaje $L(G)$ que es producida por dos o más árboles.

□ Ejemplo:

$$S \rightarrow SS \mid (S) \mid ()$$

□ Damos dos ejemplos de árboles que producen la cadena $()()()$.

Ejemplo – ()()()



Ambigüedad y Derivaciones *leftmost* y *rightmost*

- Si hay dos árboles sintácticos distintos, deben producir dos derivaciones *leftmost* diferentes, debido a la construcción dada en la prueba.
- Similarmente, dos derivaciones *leftmost* diferentes producen árboles distintos en la parte (2) de la prueba.
- Lo mismo ocurre con las derivaciones *rightmost*.

Ambigüedad, etc. – (2)

- Tenemos definiciones equivalentes para una gramática ambigua:
Una CFG G es ambigua si
 1. Existe una cadena w en $L(G)$ que posee dos derivaciones *leftmost* distintas.
 2. Existe una cadena w en $L(G)$ que posee dos derivaciones *rightmost* distintas.

Ambigüedad es una propiedad de las gramáticas

- Para el lenguaje de las cadenas con paréntesis-balanceados, podemos dar otra CFG, que es no ambigua:

$B \rightarrow (RB \mid \epsilon$

$R \rightarrow) \mid (RR$

B, el símbolo inicial, deriva cadenas balanceadas.

R genera cadenas que tienen uno o más paréntesis derechos que izquierdos.

Ejemplo: Gramática no ambigua

$$B \rightarrow (RB \mid \epsilon \quad R \rightarrow) \mid (RR$$

□ Ejercicio!

Construir una derivación *leftmost* para las siguientes cadenas de paréntesis balanceados:

(())()

() () ()

- Si deseamos expandir B, usar $B \rightarrow (RB$ si el siguiente símbolo es "(" y ϵ al final.
- Si deseamos expandir R, usar $R \rightarrow)$ si el siguiente símbolo es ")" and $(RR$ si es "(".

Proceso de *Parsing*

Remaining Input:

(())(



Next
symbol

Steps of leftmost
derivation:

B

$B \rightarrow (RB \mid \epsilon$

$R \rightarrow) \mid (RR$

Proceso de *Parsing*

Remaining Input:

$()()$



Next
symbol

Steps of leftmost
derivation:

B

(RB

$B \rightarrow (RB \mid \epsilon$

$R \rightarrow) \mid (RR$

Proceso de *Parsing*

Remaining Input:

))(



Next
symbol

Steps of leftmost
derivation:

B

(RB

((RRB

$B \rightarrow (RB \mid \epsilon$

$R \rightarrow) \mid (RR$

Proceso de *Parsing*

Remaining Input:

)()



Next
symbol

Steps of leftmost
derivation:

B

(RB

((RRB

(()RB

$B \rightarrow (RB \mid \epsilon$

$R \rightarrow) \mid (RR$

Proceso de *Parsing*

Remaining Input:

()



Next
symbol

Steps of leftmost
derivation:

B

(RB

((RRB

((()RB

((()))B

$B \rightarrow (RB \mid \epsilon$

$R \rightarrow) \mid (RR$

Proceso de *Parsing*

Remaining Input:

)



Next
symbol

Steps of leftmost
derivation:

B (())(RB

(RB

((RRB

(())RB

(())B

$B \rightarrow (RB \mid \epsilon$

$R \rightarrow) \mid (RR$

Proceso de *Parsing*

Remaining Input:

Steps of leftmost derivation:


Next
symbol

B (())(RB

(RB (()>()B

((RRB

(()RB

(())B

$B \rightarrow (RB \mid \epsilon$

$R \rightarrow) \mid (RR$

Proceso de *Parsing*

Remaining Input:

Steps of leftmost derivation:


Next
symbol

B (())(RB

(RB (()>()B

((RRB (()>()

(()RB

(())B

$B \rightarrow (RB \mid \epsilon$

$R \rightarrow) \mid (RR$

Gramáticas LL(1)

- Nota importante: Las gramáticas como $B \rightarrow (RB \mid \epsilon \quad R \rightarrow) \mid (RR,$ donde siempre se puede calcular la producción a usar en una derivación *leftmost* al escanear la cadena dada de izquierda a derecha y mirar solo el siguiente símbolo, se llaman LL(1).
 - LL(1) = "Leftmost derivation, left-to-right scan, one symbol of lookahead."

Gramáticas LL(1)

- Muchos lenguajes de programación poseen gramáticas LL(1).
- Las gramáticas LL(1) nunca son ambiguas.

Ambigüedad Inherente

- Sería muy útil si, para toda gramática ambigua, existiera un método para fijar o remover la ambigüedad, así como se hizo para el caso de la gramática de paréntesis-balancedos.
- Desafortunadamente, ciertas CFLs son *inherentemente ambiguas* : todas las gramáticas del lenguaje es ambigua.

Ejemplo: Ambigüedad inherente

□ $L = \{0^i1^j2^k : i = j \text{ ó } j = k\}$ es inherentemente ambiguo.

□ ¿Por qué?

Intuitivamente al menos una de las cadenas de la forma $0^n1^n2^n$ es generada por dos árboles sintácticos distintos, uno basado en verificar los 0s y 1s, y otros basado en verificar los 1s y 2s.

Ejemplo: Ambigüedad inherente

$S \rightarrow AB \mid CD$

A genera igual número de 0's y 1's

$A \rightarrow 0A1 \mid 01$

B genera cualquier número de 2's

$B \rightarrow 2B \mid 2$

C genera cualquier número de 0's

$C \rightarrow 0C \mid 0$

D genera igual número de 1's y 2's

$D \rightarrow 1D2 \mid 12$

Hay dos derivaciones diferentes para cada cadena de la forma $0^n1^n2^n$ (igual número de 0's, 1's, 2's.) *e.g.*

$S \Rightarrow AB \Rightarrow 01B \Rightarrow 012$

$S \Rightarrow CD \Rightarrow 0D \Rightarrow 012$