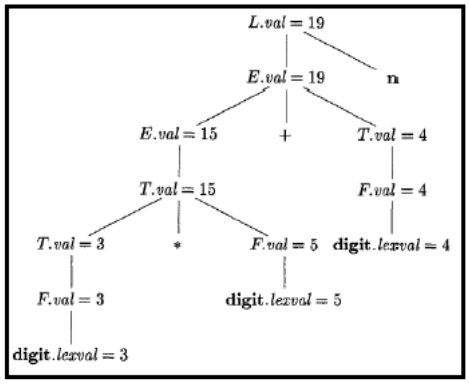
Árboles de Derivación

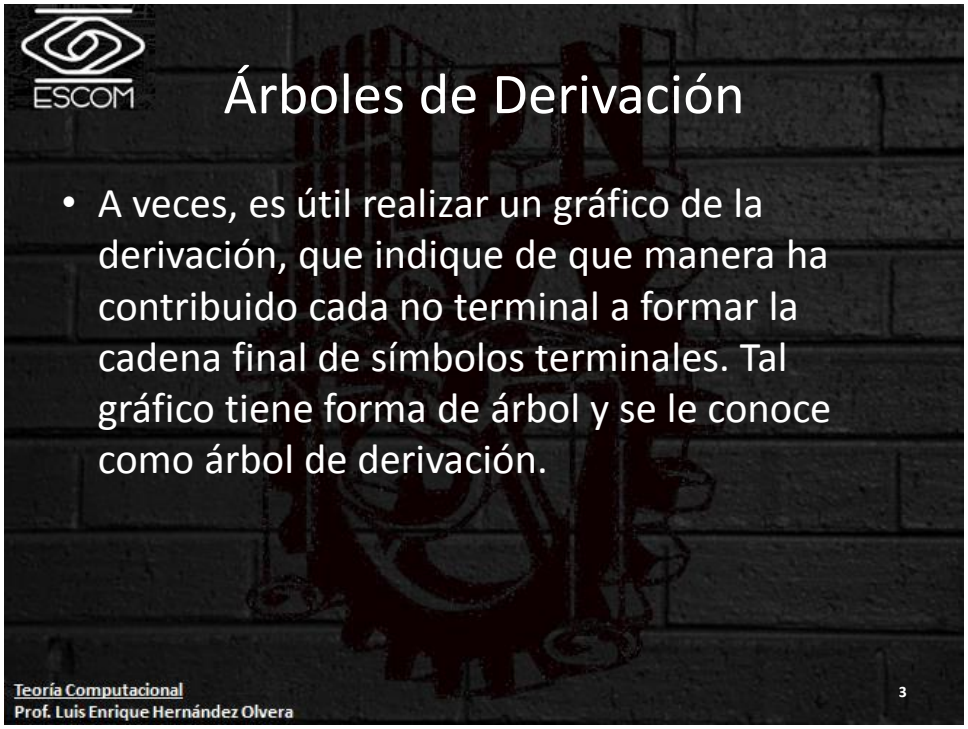
	PRODUCTION	SEMANTIC RULES
1)	$L \rightarrow E$ **n**	$L.val = E.val$
2)	$E \rightarrow E_1 + T$	$E.val = E_1.val + T.val$
3)	$E \rightarrow T$	$E.val = T.val$
4)	$T \rightarrow T_1 * F$	$T.val = T_1.val \times F.val$
5)	$T \rightarrow F$	$T.val = F.val$
6)	$F \rightarrow (E)$	$F.val = E.val$
7)	$F \rightarrow$ **digit**	$F.val =$ **digit**.lexval



Annotated parse tree for $3 * 5 + 4$ **n**

Contenido

- Árboles de Derivación.
- Gramáticas ambiguas.
- Eliminación de la ambigüedad.


ESCOM
Árboles de Derivación
• A veces, es útil realizar un gráfico de la derivación, que indique de que manera ha contribuido cada no terminal a formar la cadena final de símbolos terminales. Tal gráfico tiene forma de árbol y se le conoce como árbol de derivación.
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
3


ESCOM
Árboles de Derivación
• Los árboles de derivación están estrechamente ligados a las derivaciones y a las inferencias recursivas.
• Una de sus mas importantes aplicaciones es en la ambigüedad en las gramáticas.
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
4

Árboles de Derivación

Sea $G = (V,T,P,S)$ una gramática. Los *árboles de derivación* para G son aquellos árboles que cumplen las condiciones siguientes:

1. Cada nodo interior está etiquetado con una variable de V.
2. Cada hoja está etiquetada bien con una variable, un símbolo terminal o ε . Sin embargo, si la hoja está etiquetada con ε , entonces tiene que ser el único hijo de su padre.
3. Si un nodo interior está etiquetado como A y sus hijos están etiquetados como $X1,X2, \ldots ,Xk$ respectivamente, comenzando por la izquierda, entonces $A \longrightarrow X1X2 \cdots Xk$ es una producción de P.

Observe que el único caso en que una de las X puede reemplazarse por ε es cuando es la etiqueta del único hijo y $A \longrightarrow \varepsilon$ es una producción de G.

Árboles de Derivación Ejemplos:

Derivando ID+E de E:

Derivación de

$P \Rightarrow^* 0110$

GIC para palíndromos

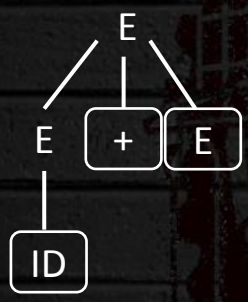
(1) P ⟶ ε
(2) P ⟶ 0
(3) P ⟶ 1
(4) P ⟶ 0P0
(5) P ⟶ 1P1

Árboles de Derivación

- Si nos fijamos en las hojas de cualquier árbol de derivación y las concatenamos empezando por la izquierda, de arriba hacia abajo, obtenemos una cadena denominada *resultado* del árbol, que siempre es una cadena que se deriva de la variable raíz.

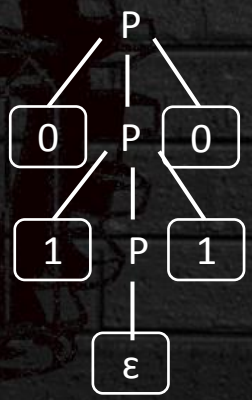
Árboles de Derivación Ejemplos:

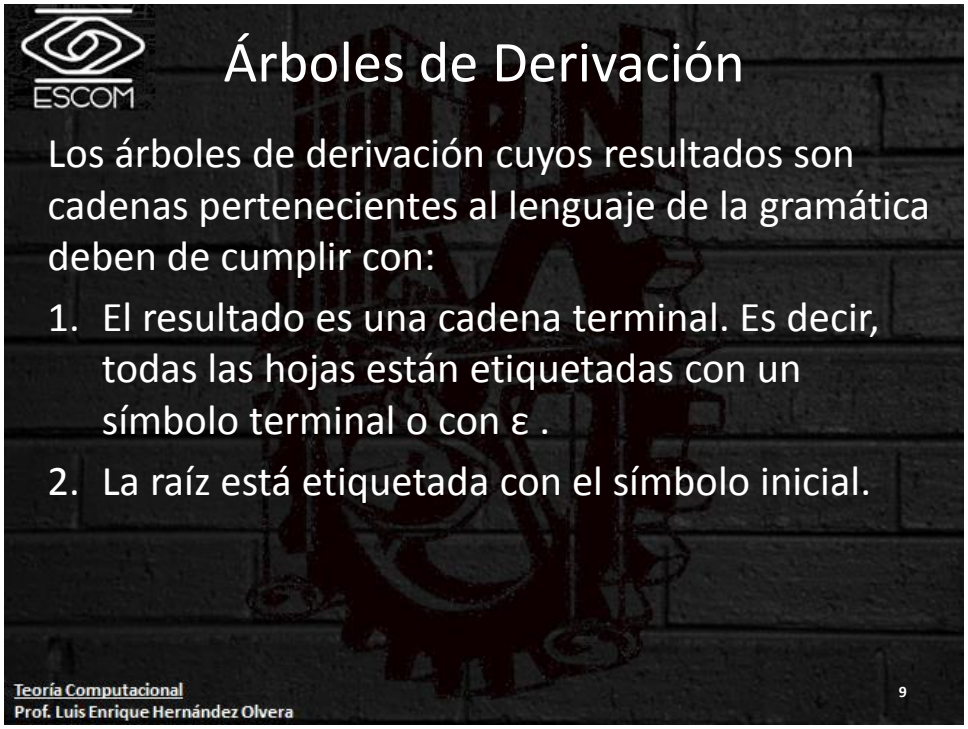
Derivando ID+E de E:

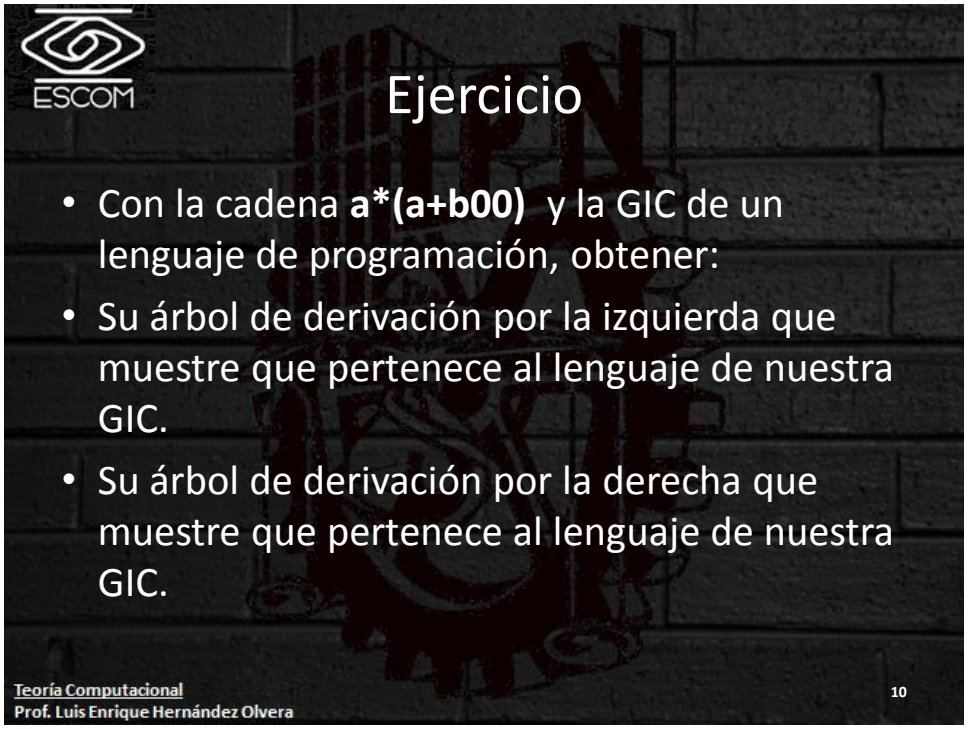


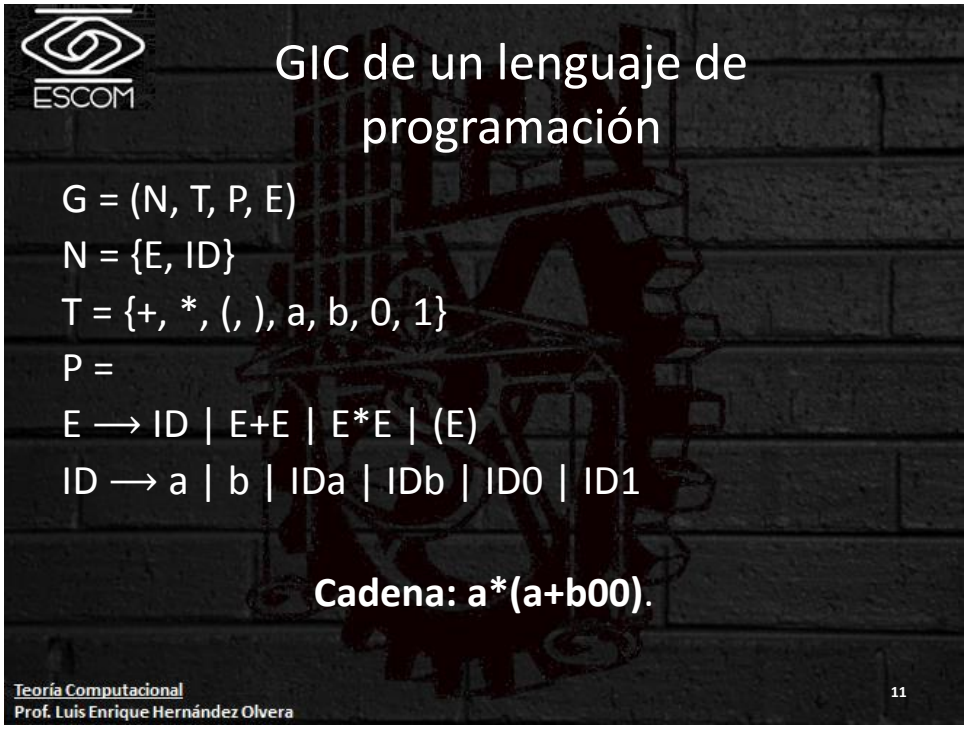
Derivación de

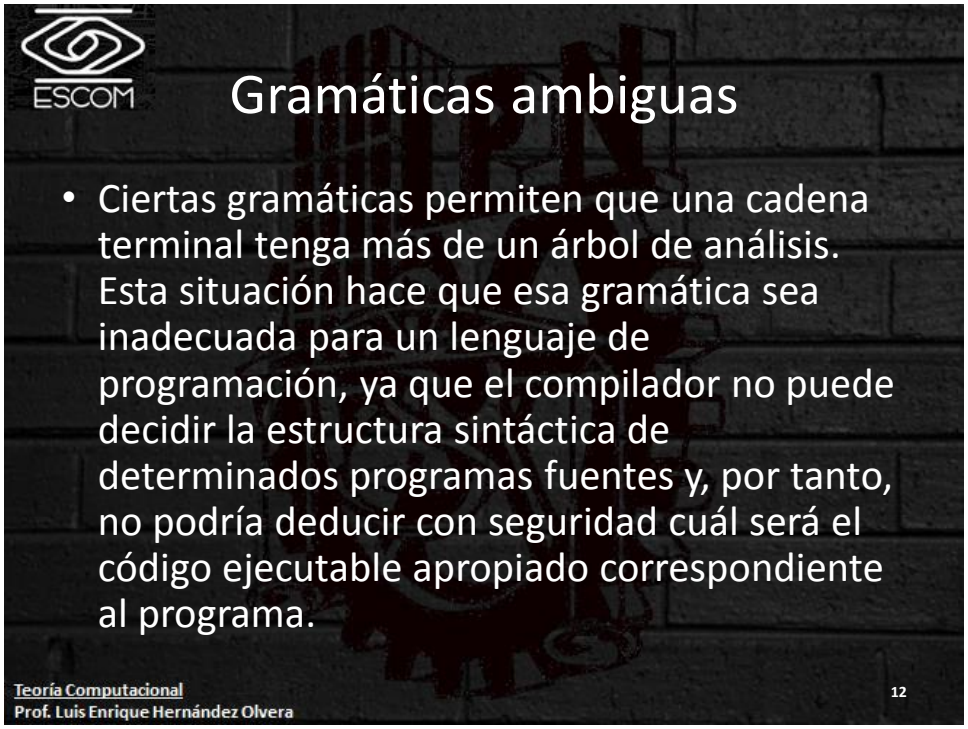
$P \Rightarrow^* 0110$

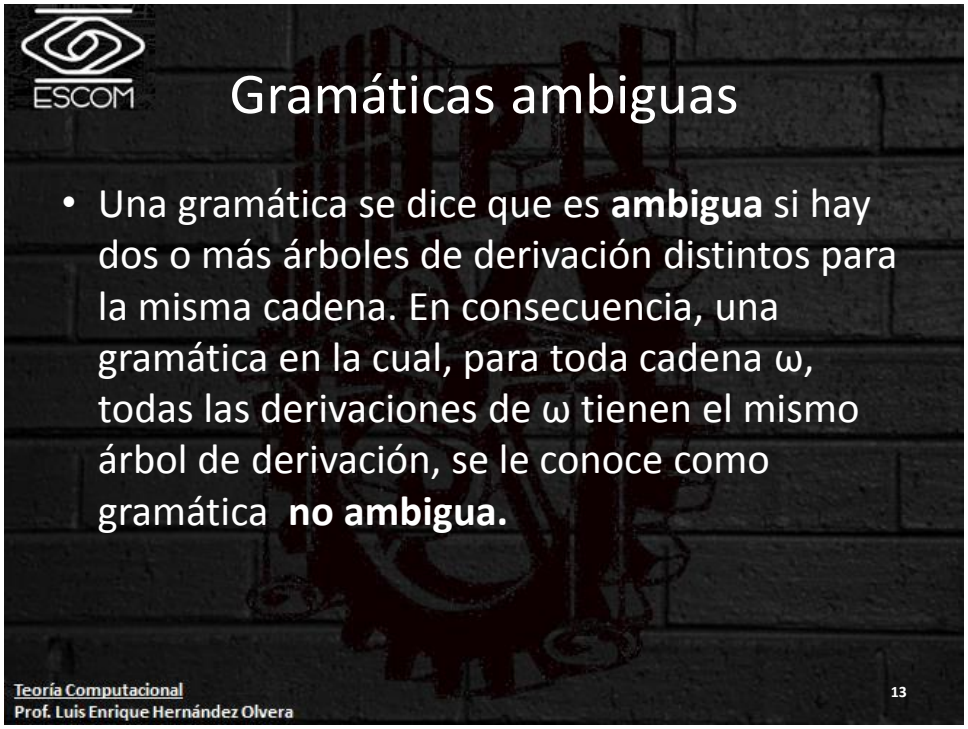



ESCOM
Árboles de Derivación
Los árboles de derivación cuyos resultados son cadenas pertenecientes al lenguaje de la gramática deben de cumplir con:
1. El resultado es una cadena terminal. Es decir, todas las hojas están etiquetadas con un símbolo terminal o con ε .
2. La raíz está etiquetada con el símbolo inicial.
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
9


ESCOM
Ejercicio
• Con la cadena **a*(a+b00)** y la GIC de un lenguaje de programación, obtener:
• Su árbol de derivación por la izquierda que muestre que pertenece al lenguaje de nuestra GIC.
• Su árbol de derivación por la derecha que muestre que pertenece al lenguaje de nuestra GIC.
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
10

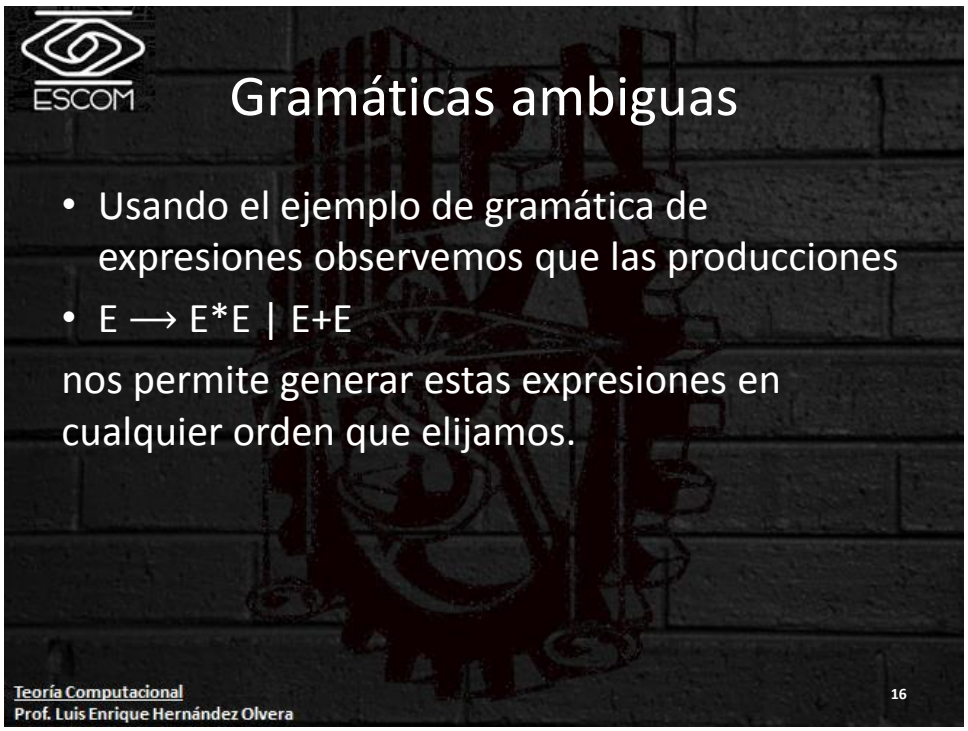

ESCOM
GIC de un lenguaje de programación
G = (N, T, P, E)
N = {E, ID}
T = {+, *, (,), a, b, 0, 1}
P =
E ⟶ ID | E+E | E*E | (E)
ID ⟶ a | b | IDa | IDb | ID0 | ID1
Cadena: a*(a+b00).
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
11


ESCOM
Gramáticas ambiguas
• Ciertas gramáticas permiten que una cadena terminal tenga más de un árbol de análisis. Esta situación hace que esa gramática sea inadecuada para un lenguaje de programación, ya que el compilador no puede decidir la estructura sintáctica de determinados programas fuentes y, por tanto, no podría deducir con seguridad cuál será el código ejecutable apropiado correspondiente al programa.
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
12


ESCOM
Gramáticas ambiguas
• Una gramática se dice que es **ambigua** si hay dos o más árboles de derivación distintos para la misma cadena. En consecuencia, una gramática en la cual, para toda cadena ω, todas las derivaciones de ω tienen el mismo árbol de derivación, se le conoce como gramática **no ambigua.**
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
13


ESCOM
Tipos de ambigüedad
Dentro del estudio de gramáticas existen dos tipos fundamentales de ambigüedad, los cuales son:
Ambigüedad Inherente
Ambigüedad Transitoria
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
14

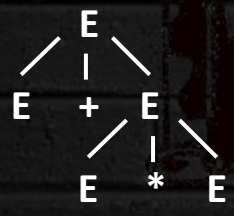

ESCOM
Tipos de ambigüedad
• **Ambigüedad Inherente**: Las gramáticas que presentan este tipo de ambigüedad no pueden utilizarse para lenguajes de programación, ya que por más transformaciones que se realicen sobre ellas, NUNCA se podrá eliminar completamente la ambigüedad que presentan.
• **Ambigüedad Transitoria**: La ambigüedad puede llegar a ser eliminada realizando una serie de transformaciones sobre la gramática original.
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
15


ESCOM
Gramáticas ambiguas
• Usando el ejemplo de gramática de expresiones observemos que las producciones
• E ⟶ E*E | E+E
nos permite generar estas expresiones en cualquier orden que elijamos.
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
16

Gramáticas ambiguas
Ejemplo

- consideremos la forma sentencial E+E∗E. Existen dos derivaciones de E:

(1) E ⟶ E+E



(2) E ⟶ E*E

Gramáticas ambiguas
Ejemplo

- consideremos la cadena a+a∗a. Existen dos derivaciones de E *⇒ a+a*a:

1)

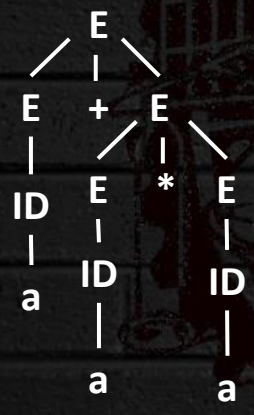
E ⇒ E+E ⇒ ID+E ⇒ a+E ⇒ a+E*E ⇒ a+ID*E ⇒ a+a*E ⇒ a+a*ID ⇒ a+a*a

2)

E ⇒ E*E ⇒ E+E*E ⇒ ID+E*E ⇒ a+E*E ⇒ a+ID*E ⇒ a+a*E ⇒ a+a*ID ⇒ a+a*a

Gramáticas ambiguas
Ejemplo

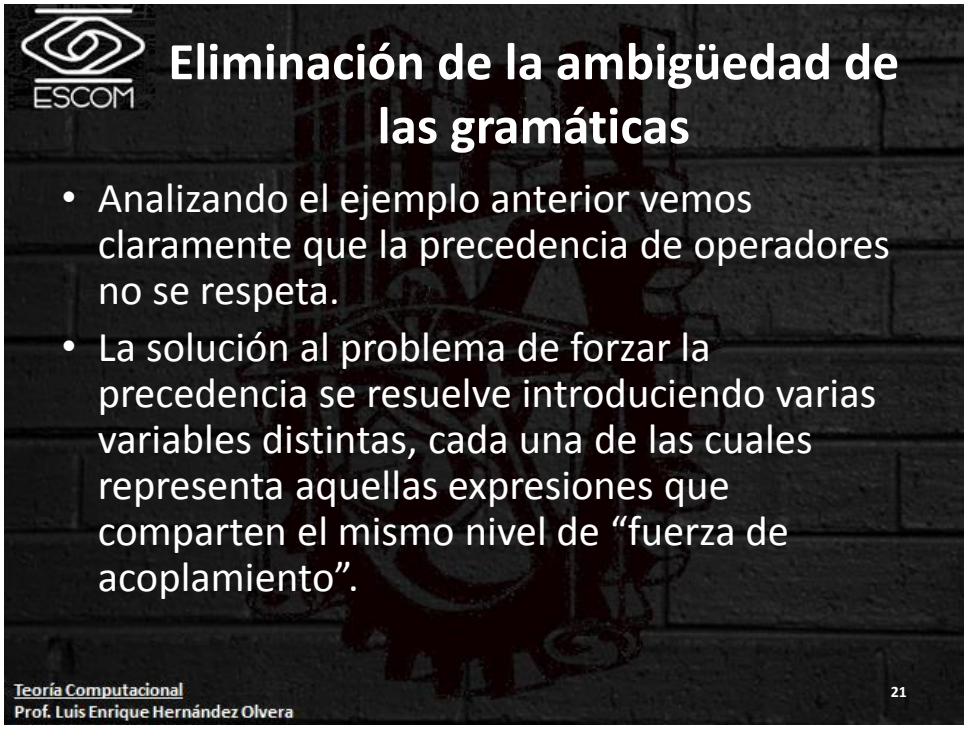
(1) E ⟶ E+E

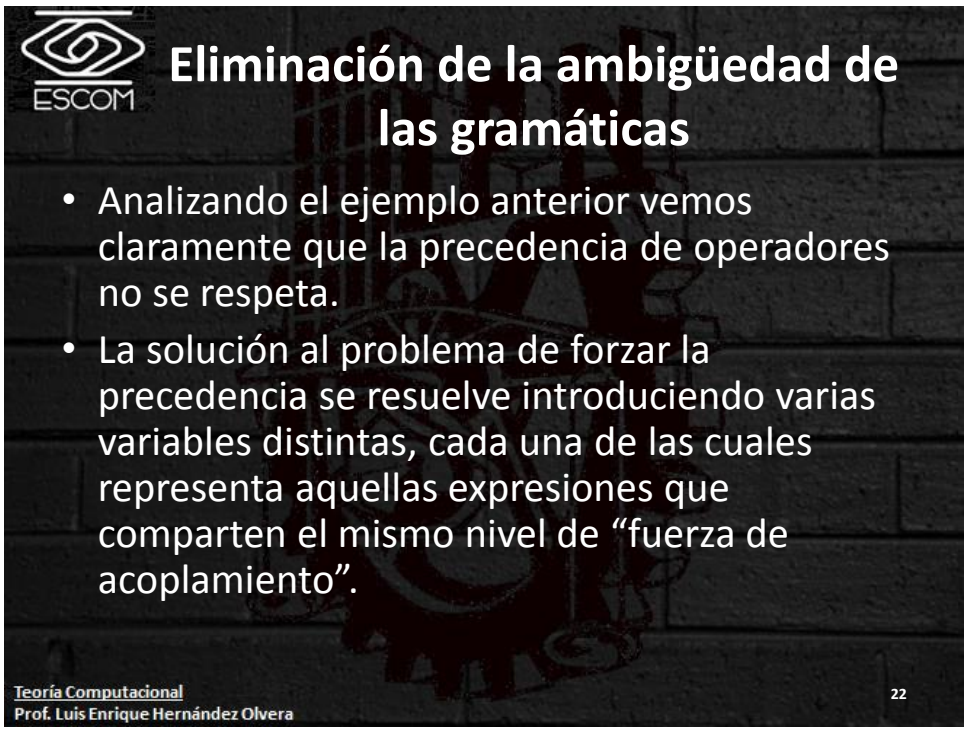


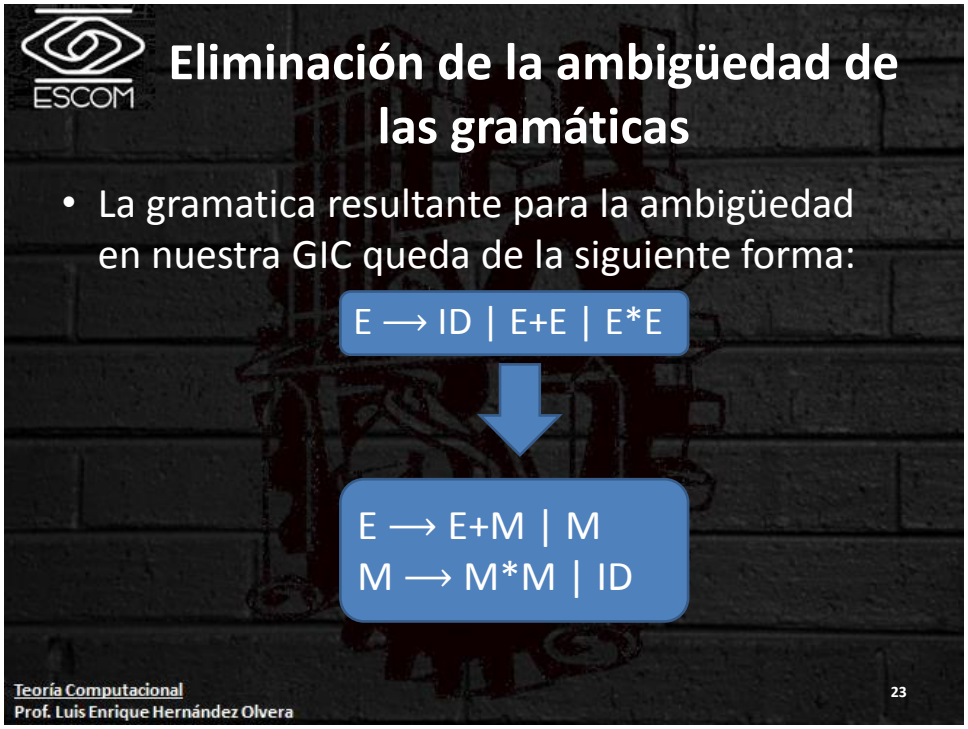
(2) E ⟶ E*E

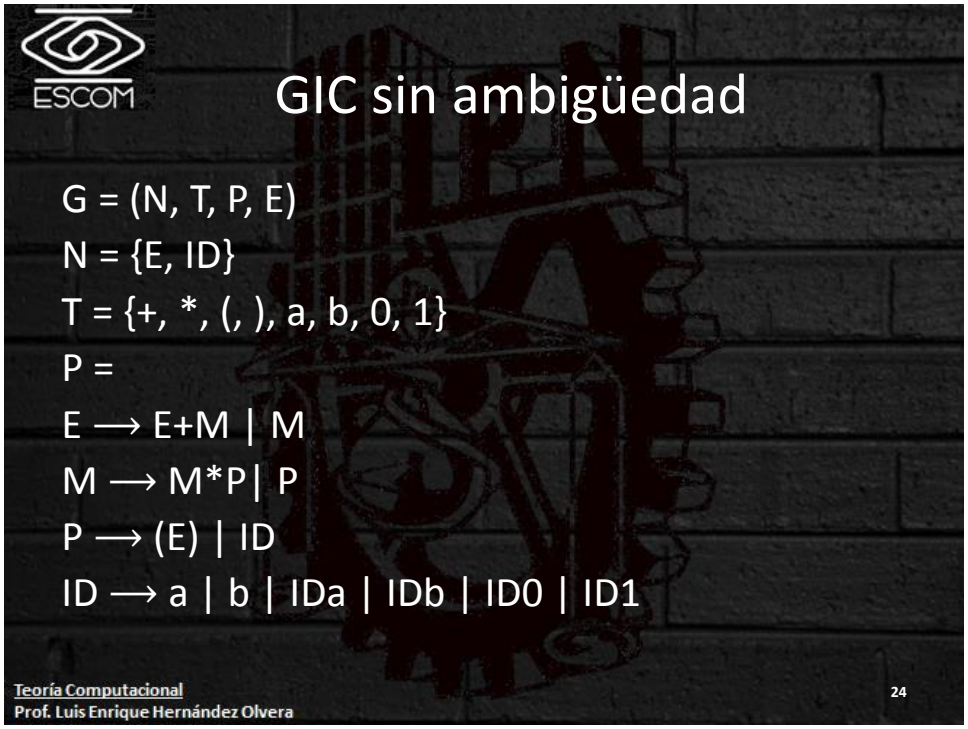
Eliminación de la ambigüedad de las gramáticas

- no existe un algoritmo que nos diga si una GIC es ambigua.
- Para los tipos de construcciones que aparecen en los lenguajes de programación comunes, existen técnicas bien conocidas que permiten eliminar la ambigüedad.


ESCOM
Eliminación de la ambigüedad de las gramáticas
• Analizando el ejemplo anterior vemos claramente que la precedencia de operadores no se respeta.
• La solución al problema de forzar la precedencia se resuelve introduciendo varias variables distintas, cada una de las cuales representa aquellas expresiones que comparten el mismo nivel de "fuerza de acoplamiento".
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
21


ESCOM
Eliminación de la ambigüedad de las gramáticas
• Analizando el ejemplo anterior vemos claramente que la precedencia de operadores no se respeta.
• La solución al problema de forzar la precedencia se resuelve introduciendo varias variables distintas, cada una de las cuales representa aquellas expresiones que comparten el mismo nivel de "fuerza de acoplamiento".
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
22


ESCOM
Eliminación de la ambigüedad de las gramáticas
• La gramatica resultante para la ambigüedad en nuestra GIC queda de la siguiente forma:
E ⟶ ID | E+E | E*E
E ⟶ E+M | M
M ⟶ M*M | ID
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
23


ESCOM
GIC sin ambigüedad
G = (N, T, P, E)
N = {E, ID}
T = {+, *, (,), a, b, 0, 1}
P =
E ⟶ E+M | M
M ⟶ M*P| P
P ⟶ (E) | ID
ID ⟶ a | b | IDa | IDb | ID0 | ID1
Teoría Computacional
Prof. Luis Enrique Hernández Olvera
24