# Tema 4:
# Gramáticas independientes del contexto

Teoría de autómatas y lenguajes formales I

# Bibliografía


- Hopcroft, J. E., Motwani, R., y Ullman, J. D. "Introducción a la Teoría de Autómatas, Lenguajes y Computación". Addison Wesley. 2002.
  - capítulo 5
- Sudkamp, Thomas A. "Languages and machines : an introduction to the theory of computer science". Addison-Wesley Publishing Company, 1998.
  - capítulos 3 y 4

# Introducción

- Lenguajes independientes del contexto (LIC)
- Gramáticas independientes del contexto (GIC)
  - implementación de analizadores sintácticos
  - descripción de formatos de documentos: XML
- Árboles sintácticos: representan gráficamente la estructura de la gramática
- Autómatas con pila

# Un ejemplo de GIC

- Palíndromo: $w = w^R$
- Este lenguaje no es regular. Se comprueba aplicando el lema del bombeo a la cadena $0^n10^n$
- Para generar el lenguaje de los palíndromos con el alfabeto {0, 1}:
  - Base: ε, 0, 1 son palíndromos
  - Paso inductivo: si *w* es palíndromo, también lo son 0*w*0 y 1*w*1
- Reglas que definen la gramática que genera el lenguaje de los palíndromos con alfabeto {0, 1}:
  - $P \longrightarrow \varepsilon$
  - $P \longrightarrow 0$
  - $P \longrightarrow 1$
  - $P \longrightarrow 0P0$
  - $P \longrightarrow 1P1$
  - En notación compacta: $P \longrightarrow \varepsilon \mid 0 \mid 1 \mid 0P0 \mid 1P1$

# Definición de gramática

- Las GIC están formadas por cuatro componentes:
  - el alfabeto de símbolos terminales, que son los símbolos finales del lenguaje
  - el conjunto finito de símbolos no terminales o variables, que permiten representar subconjuntos del lenguaje o estados intermedios en la generación de las palabras del lenguaje
  - el símbolo inicial o axioma de la gramática (una de las variables), a partir de cual se obtiene cualquier palabra del lenguaje
  - un conjunto finito de producciones o reglas, que indican las transformaciones posibles desde los símbolos no terminales a las palabras del lenguaje. Las reglas están formadas por:
    - una variable, cabeza de la producción
    - el símbolo de producción $\longrightarrow$
    - una cadena de 0 o más símbolos terminales y variables, que son el cuerpo de la producción
    - las producciones son de la forma $B \longrightarrow x$, donde $B \in V$, y $x \in (V \cup T)^*$

- $G = (V, T, P, S)$
  - ejemplo: $G_{palíndromo} = (\{P\}, \{0, 1\}, A, P)$, donde $A$ son las producciones o reglas: $P \longrightarrow \varepsilon \mid 0 \mid 1 \mid 0P0 \mid 1P1$

# Definición de gramática (II)

- Ejemplo: $G = (\{E, I\}, \{+, *, (, ), a, b, 0, 1\}, P, E)$, donde $P$ es el conjunto de producciones:

$$
\begin{array}{lrcl}
1. & E & \rightarrow & I \\
2. & E & \rightarrow & E + E \\
3. & E & \rightarrow & E * E \\
4. & E & \rightarrow & (E) \\
5. & I & \rightarrow & a \\
6. & I & \rightarrow & b \\
7. & I & \rightarrow & Ia \\
8. & I & \rightarrow & Ib \\
9. & I & \rightarrow & I0 \\
10. & I & \rightarrow & I1
\end{array}
$$

# Derivaciones de una gramática

- Aplicando las producciones de una GIC se infieren las cadenas que están en el lenguaje
- Sea $G = (V, T, P, S)$, y $\alpha A \beta$ una cadena de terminales y variables, donde $A$ está en $V$, y $\alpha$ y $\beta$ están en $(V \cup T)^*$. Sea $A \longrightarrow \gamma$ una producción de $G$. Entonces:

$$\alpha A \beta \underset{G}{\Rightarrow} \alpha \gamma \beta$$

- Extensión a varios pasos de derivación
  - si $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$ y $\beta \underset{G}{\Rightarrow} \gamma$, entonces $\alpha \overset{*}{\underset{G}{\Rightarrow}} \gamma$
- Ejemplo: inferencia de “a * (a + b00)”

$$E \Rightarrow E * E \Rightarrow I * E \Rightarrow a * E \Rightarrow$$

$$a * (E) \Rightarrow a * (E + E) \Rightarrow a * (I + E) \Rightarrow a * (a + E) \Rightarrow$$

$$a * (a + I) \Rightarrow a * (a + I0) \Rightarrow a * (a + I00) \Rightarrow a * (a + b00)$$

# Derivaciones izquierda y derecha

- Derivación más a la izquierda: $\overset{*}{\underset{mi}{\Rightarrow}}$
- Derivación más a la derecha: $\overset{*}{\underset{md}{\Rightarrow}}$
- Ejemplo:

$$E \overset{*}{\underset{mi}{\Rightarrow}} a*(a+b00)$$

$$E \underset{mi}{\Rightarrow} E*E \underset{mi}{\Rightarrow} I*E \underset{mi}{\Rightarrow} a*E \underset{mi}{\Rightarrow}$$

$$a*(E) \underset{mi}{\Rightarrow} a*(E+E) \underset{mi}{\Rightarrow} a*(I+E) \underset{mi}{\Rightarrow} a*(a+E) \underset{mi}{\Rightarrow}$$

$$a*(a+I) \underset{mi}{\Rightarrow} a*(a+I0) \underset{mi}{\Rightarrow} a*(a+I00) \underset{mi}{\Rightarrow} a*(a+b00)$$

$$E \overset{*}{\underset{md}{\Rightarrow}} a*(a+b00)$$

$$E \underset{md}{\Rightarrow} E*E \underset{md}{\Rightarrow} E*(E) \underset{md}{\Rightarrow} E*(E+E) \underset{md}{\Rightarrow}$$

$$E*(E+I) \underset{md}{\Rightarrow} E*(E+I0) \underset{md}{\Rightarrow} E*(E+I00) \underset{md}{\Rightarrow} E*(E+b00) \underset{md}{\Rightarrow}$$

$$E*(I+b00) \underset{md}{\Rightarrow} E*(a+b00) \underset{md}{\Rightarrow} I*(a+b00) \underset{md}{\Rightarrow} a*(a+b00)$$

# Lenguaje de una gramática

- Si $G = (V, T, P, S)$ es una GIC, el lenguaje de $G$ será:
  - $L(G) = \{w \text{ que están en } T^* \mid S \overset{*}{\underset{G}{\Rightarrow}} w \}$

- Notación:
  - $a, b$, ..., +, (, ...: símbolos terminales
  - $A, B$, ...: variables
  - ..., $w, x, y, z$: cadenas de terminales
  - ..., $X, Y, Z$: terminales o variables
  - $\alpha, \beta, \gamma$, ...: cadenas de terminales y variables

# Formas sentenciales

- Si $G = (V, T, P, S)$ es una GIC, cualquier cadena $\alpha$ en $(V \cup T)^*$ tal que $S \overset{*}{\Rightarrow} \alpha$ es una forma sentencial
- El lenguaje $L(G)$ está formado por las formas sentenciales que están en $T^*$: se denominan sentencias
- Ejemplo:

$$E \Rightarrow E * E \Rightarrow E * (E) \Rightarrow E * (E + E) \Rightarrow E * (I + E)$$

$$E \underset{mi}{\Rightarrow} E * E \underset{mi}{\Rightarrow} I * E \underset{mi}{\Rightarrow} a * E$$

$$E \underset{md}{\Rightarrow} E * E \underset{md}{\Rightarrow} E * (E) \underset{md}{\Rightarrow} E * (E + E)$$

# Problemas

1. Diseñar las GIC para los siguientes lenguajes
   1. $\{0^n1^n \mid n \geq 1\}$
   2. $\{a^ib^jc^k \mid i \neq j \quad \text{o} \quad j \neq k\}$
2. La siguiente gramática genera el lenguaje de expresiones regulares $0^*1(0+1)^*$: $S \rightarrow A1B$, $A \rightarrow 0A \mid \varepsilon$, $B \rightarrow 0B \mid 1B \mid \varepsilon$. Obtener las derivaciones más a la izquierda y más a la derecha para la cadena: 00101
3. Sea $T = \{0, 1, (, ), +, *, \varnothing, e\}$. Construir una GIC con el conjunto de terminales $T$ que genere expresiones regulares válidas con el alfabeto {0, 1} (considerar $e$ como $\varepsilon$).

# Árboles de derivación

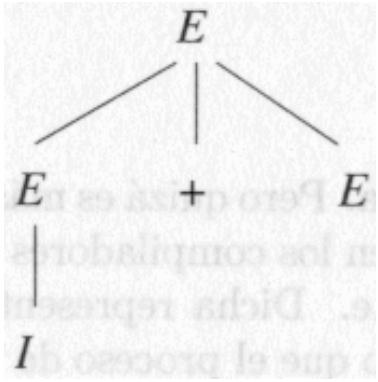
- Representación de las derivaciones en forma de árbol
  - muestra la agrupación de los símbolos de una cadena terminal en subcadenas
  - utilización en los compiladores para representar la estructura sintáctica del programa fuente

- Terminología
  - los árboles son conjuntos de nodos unidos entre sí por la relación *padre-hijo*. Un nodo tiene como máximo un padre y cero o más hijos
  - nodo raíz: no tiene padre
  - hojas: nodos sin hijos
  - nodos interiores: nodos que no son hojas
  - el hijo de un hijo de ...: descendiente
  - el padre de un padre de ...: antepasado
  - los hijos de un nodo se ordenan de izquierda a derecha

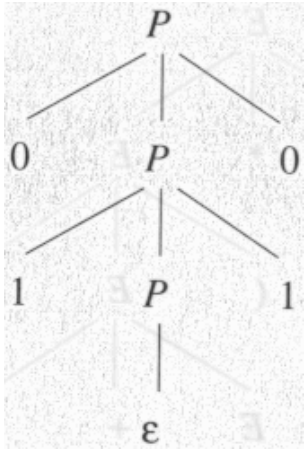
# Construcción de árboles de derivación

- Sea $G = (V, T, P, S)$. El árbol de derivación para $G$ tendrá las siguientes características:
  - cada nodo interior está etiquetado con una variable
  - cada hoja está etiquetada con una variable, un terminal o ε. Si es ε, tiene que ser el único hijo de su padre
  - si un nodo interior está etiquetado con $A$, y sus hijos están etiquetados con $X_1, X_2, ..., X_k$ (de izquierda a derecha), entonces $A \longrightarrow X_1X_2...X_k$ es una producción de $P$
- Ejemplos:

$$E \overset{*}{\Rightarrow} I + E$$




$$P \overset{*}{\Rightarrow} 0110$$



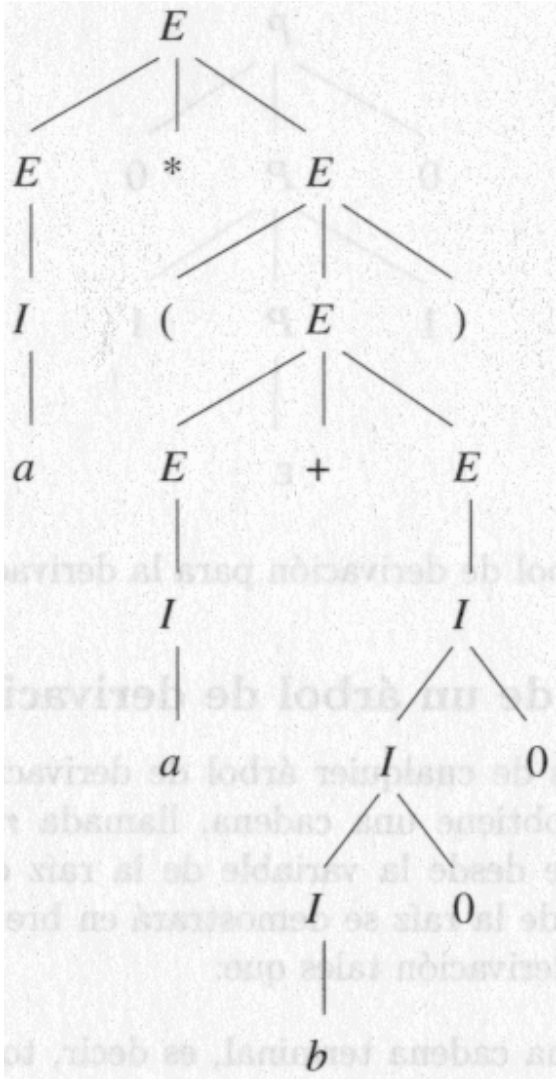
# Resultado de un árbol de derivación

- Concatenando la hojas de un árbol desde la izquierda se obtiene la cadena resultado del árbol, que se deriva desde la raíz
- Son especialmente importantes los árboles de derivación tales que:
  - su resultado es una cadena terminal
  - la raíz está etiquetada con el símbolo inicial
- Estos árboles tienen como resultado las cadenas del lenguaje generado por esa gramática

# Un ejemplo

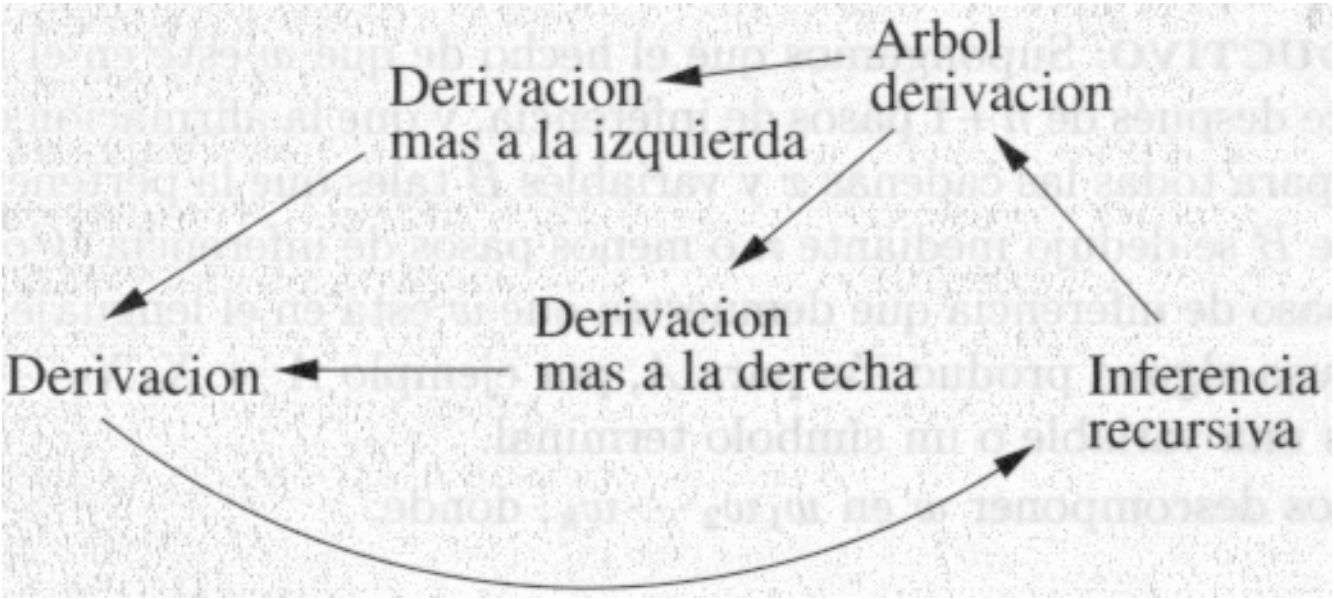
$$E \overset{*}{\Rightarrow} a*(a+b00)$$

| | | | |
|---|---|---|---|
| 1. | $E$ | $\rightarrow$ | $I$ |
| 2. | $E$ | $\rightarrow$ | $E+E$ |
| 3. | $E$ | $\rightarrow$ | $E*E$ |
| 4. | $E$ | $\rightarrow$ | $(E)$ |
| 5. | $I$ | $\rightarrow$ | $a$ |
| 6. | $I$ | $\rightarrow$ | $b$ |
| 7. | $I$ | $\rightarrow$ | $Ia$ |
| 8. | $I$ | $\rightarrow$ | $Ib$ |
| 9. | $I$ | $\rightarrow$ | $I0$ |
| 10. | $I$ | $\rightarrow$ | $I1$ |



# Inferencia, derivaciones y árboles

- Dada $G = (V, T, P, S)$, las siguientes afirmaciones son equivalentes:
  - el procedimiento de inferencia recursiva determina que la cadena terminal $w$ pertenece al lenguaje de la variable $A$
  - $A \overset{*}{\Rightarrow} w$
  - $A \overset{*}{\underset{mi}{\Rightarrow}} w$
  - $A \overset{*}{\underset{md}{\Rightarrow}} w$

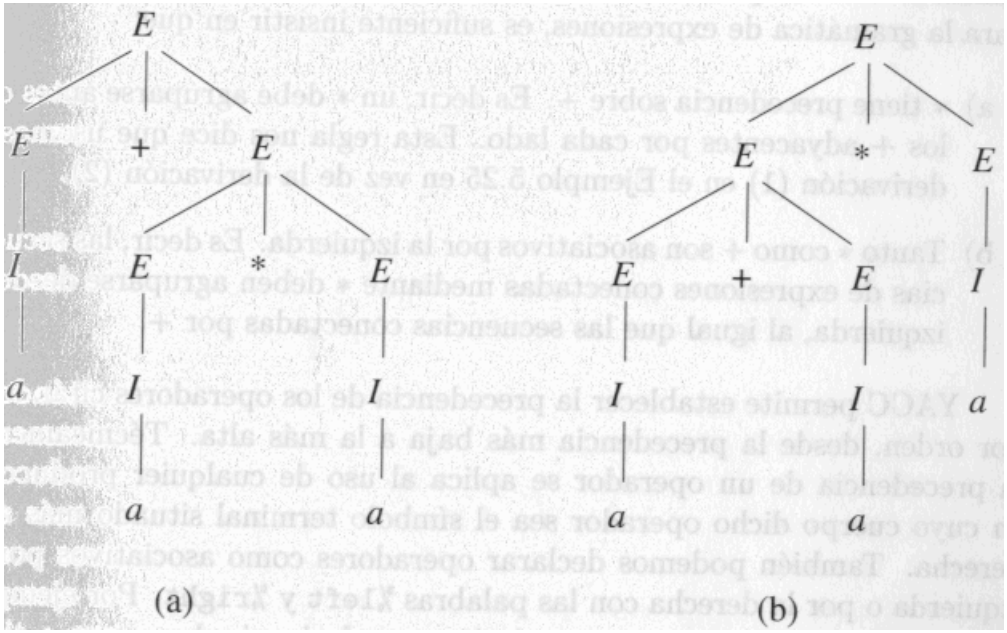



  - Existe un árbol de derivación con la raíz etiquetada con $A$ y resultado $w$

# Ambigüedad

- Una GIC $G = (V, T, P, S)$ es ambigua si existe al menos una cadena $w$ en $T^*$ para la que podemos encontrar dos árboles de derivación distintos con la raíz etiquetada con $S$ y cuyo resultado es $w$

1. $E \rightarrow I$
2. $E \rightarrow E + E$
3. $E \rightarrow E * E$
4. $E \rightarrow (E)$
5. $I \rightarrow a$
6. $I \rightarrow b$
7. $I \rightarrow Ia$
8. $I \rightarrow Ib$
9. $I \rightarrow I0$
10. $I \rightarrow I1$



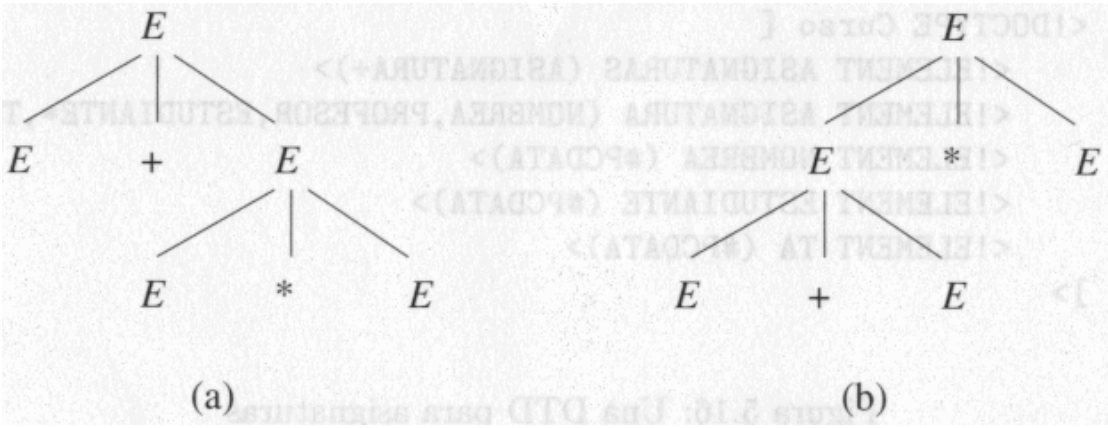

- La existencia de derivaciones diferentes para una cadena no supone un defecto en la gramática, *la existencia de árboles de derivación diferentes sí supone un problema*

  - $E \rightarrow E + E \rightarrow I + E \rightarrow a + E \rightarrow a + I \rightarrow a + b$
  - $E \rightarrow E + E \rightarrow E + I \rightarrow E + b \rightarrow I + b \rightarrow a + b$

# Eliminación de la ambigüedad

- No existe un algoritmo que nos indique si una GIC es ambigua
- Existen LIC que sólo tienen GIC ambiguas: inherentemente ambiguos
- Para las construcciones de los lenguajes de programación comunes existen técnicas para la eliminación de la ambigüedad
- Ejemplo: causas de ambigüedad en la siguiente gramática
  - no se respeta la precedencia de operadores
  - una secuencia de operadores idénticos puede agruparse desde la izquierda y desde la derecha. Lo convencional es agrupar desde la izquierda

1. $E \rightarrow I$
2. $E \rightarrow E + E$
3. $E \rightarrow E * E$
4. $E \rightarrow (E)$
5. $I \rightarrow a$
6. $I \rightarrow b$
7. $I \rightarrow Ia$
8. $I \rightarrow Ib$
9. $I \rightarrow I0$
10. $I \rightarrow I1$

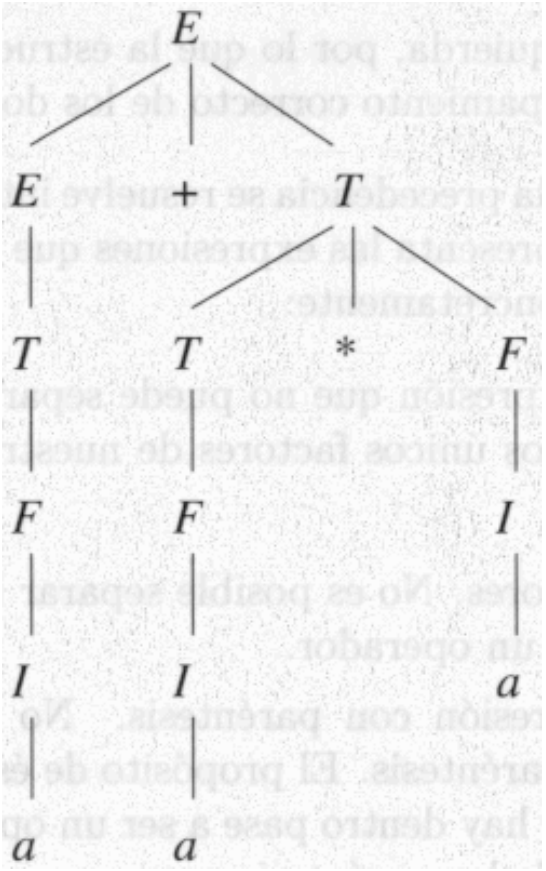


# Eliminación de la ambigüedad (II)

- Ejemplo: modificamos la gramática para forzar la precedencia

1. $E \rightarrow I$
2. $E \rightarrow E + E$
3. $E \rightarrow E * E$
4. $E \rightarrow (E)$
5. $I \rightarrow a$
6. $I \rightarrow b$
7. $I \rightarrow Ia$
8. $I \rightarrow Ib$
9. $I \rightarrow I0$
10. $I \rightarrow I1$

$$
\begin{aligned}
I &\rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \\
F &\rightarrow I \mid (E) \\
T &\rightarrow F \mid T * F \\
E &\rightarrow T \mid E + T
\end{aligned}
$$



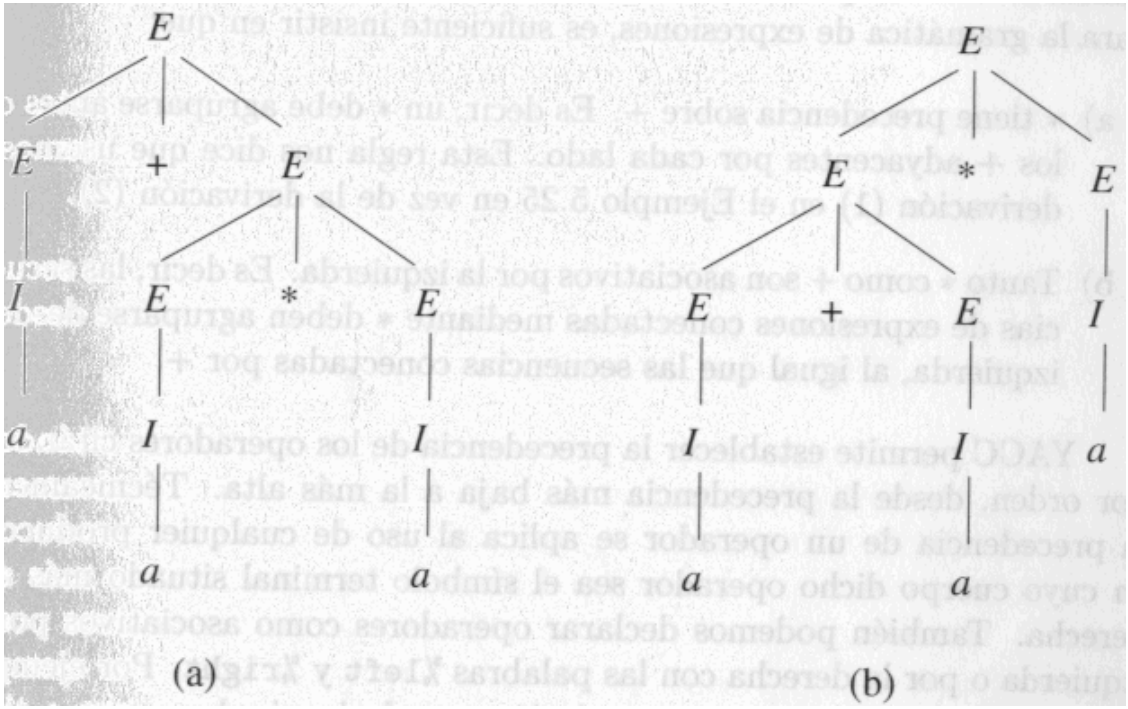
# Derivaciones como forma de expresar ambigüedad

- En una gramática no ambigua las derivaciones más a la derecha y más a la izquierda serán únicas

- Teorema: para toda gramática $G = (V, T, P, S)$ y toda cadena $w$ en $T^*$, $w$ tiene dos árboles de derivación distintos si y sólo si $w$ tiene dos derivaciones más a la izquierda distintas desde $S$
  - lo mismo para las derivaciones más a la derecha

# Derivaciones como forma de expresar ambigüedad (II)

- Ejemplo:




a) $E \underset{mi}{\Rightarrow} E+E \underset{mi}{\Rightarrow} I+E \underset{mi}{\Rightarrow} a+E \underset{mi}{\Rightarrow} a+E*E \underset{mi}{\Rightarrow} a+I*E \underset{mi}{\Rightarrow} a+a*E \underset{mi}{\Rightarrow} a+a*I \underset{mi}{\Rightarrow} a+a*a$

b) $E \underset{mi}{\Rightarrow} E*E \underset{mi}{\Rightarrow} E+E*E \underset{mi}{\Rightarrow} I+E*E \underset{mi}{\Rightarrow} a+E*E \underset{mi}{\Rightarrow} a+I*E \underset{mi}{\Rightarrow} a+a*E \underset{mi}{\Rightarrow} a+a*I \underset{mi}{\Rightarrow} a+a*a$

# Problemas finales

1. Un palíndromo es una cadena que se lee igual de izquierda a derecha que de derecha a izquierda. Por ejemplo, las palabras “radar”, “oso” y “abba” son palíndromos. Dado el alfabeto $\Sigma = \{a, b\}$, determinar una gramática que describa palíndromos. La gramática debería generar palabras como “abba”, “aba”, “bb”, “babab”, “a”, “b”, ..., y $\varepsilon$.
2. Determinar una gramática que genere el lenguaje $L = \{a^i b^j c^k \mid i=j \text{ o } j=k, i, j, k > 0\}$.
3. Determinar una gramática que genere el lenguaje $L = \{a^i b^i a^j b^j \mid i, j > 0\}$,