FUNDAMENTOS DE LA TEORÍA DE GRAMÁTICAS FORMALES

Lenguajes Formales - Facultad de Ingenieria- UNJu
Ing Fabiana R. Aragón

DEFINICIÓN DE PRODUCCIÓN O REGLA

Sea Σ un Alfabeto

Se llama producción o regla a un ***par ordenado*** *(x, y), donde x e y* $\in \Sigma^*$

x ::= y

DEFINICIÓN DE DERIVACIÓN

- **DERIVACIÓN DIRECTA *v* → *w***

Es la aplicación de la producción x ::= y a una palabra v para convertirla en otra w.

Σ ={*a, b, c*};
a::=*ba*
v y w son dos palabras sobre Σ

- *v* = *aa*
  ba
- *w* = *baa*

DEFINICIÓN DE DERIVACIÓN

- **DERIVACIÓN $v \rightarrow {}^{*}w$**

Es la aplicación de una secuencia de producciones a una palabra.

$\Sigma = \{a, b, c\}$;
$a ::= ba$
v y w son dos palabras sobre Σ

- $v = aa$

ba

- $w = baa$
- $w_1 = bbaa$
- $w_2 = bbbaaa$

TIPOS DE DERIVACIÓN

- **DERIVACIÓN MÁS A LA IZQUIERDA (DMI)**

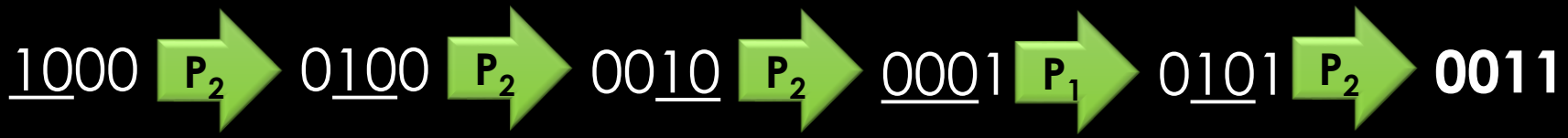
Si se utiliza en cada derivación directa la producción aplicada a los símbolos más a la izquierda de la palabra.

Ejemplo:

P_1 : (000 ::= 010)

P_2 : (10 ::= 01)

La derivación más a la Izquierda queda:



TIPOS DE DERIVACIÓN

- **DERIVACIÓN MÁS A LA DERECHA (DMD)**

Si se utiliza en cada derivación directa la producción aplicada a los símbolos más a la derecha de la palabra.

Ejemplo:

P_1 : (000 ::= 010)

P_2 : (10 ::= 01)

La derivación más a la derecha queda:

1000 $\xrightarrow{P_1}$ 10<u>10</u> $\xrightarrow{P_2}$ <u>10</u>01 $\xrightarrow{P_2}$ 0<u>10</u>1 $\xrightarrow{P_1}$ **0011**

DEFINICIÓN DE GRAMÁTICA

GRAMATICA: es un método para la generación de palabras de un lenguaje a partir de un alfabeto.

CONCEPTO DE GRAMÁTICA FORMAL

Se denomina *gramática formal a la cuádrupla:*

Σ_N, alfabeto de símbolos No Terminales

Σ_T, alfabeto de símbolos Terminales

$$G = (\Sigma_T, \Sigma_N, S, P)$$

Se verifica además que:
- $\Sigma_T \cap \Sigma_N = \emptyset$
- *el alfabeto es* $\Sigma = \Sigma_T \cup \Sigma_N$

$S \in \Sigma_N$, es el axioma o símbolo inicial

P es un conjunto finito de reglas de producción de la forma $u ::= v$, *donde* $u \in \Sigma^+$ *y* $v \in \Sigma^*$.

CONCEPTO DE GRAMÁTICA FORMAL

Forma normal de Backus

- Ejemplo: consideremos la gramática
  Σ_T = {0, 1, 2}
  Σ_N = {N, C}
  S = N
  P = { N::=NC, N::=C, C::=0,
  C::=1, C::=2}
- Una notación simplificada para las reglas de producción. Si existen dos reglas de la forma
  - *u*::=*v*
  - *u*::=*w*

  se pueden representar de la forma:
  - ***u*::=*v* | *w***

Formas sentenciales y sentencias

La forma sentencial de G es si se verifica que S→* x
Las siguientes son formas sentenciales: **NCC, NC2, 120**
S = N→NC → NCC
S = N → NC → NCC → NC2
S = N → NC → NCC → CCC → 1CC → 12C → 120

se dice que x es una sentencia o instrucción de G. Es decir, las sentencias estarán compuestas únicamente por símbolos terminales.

LENGUAJE ASOCIADO A UNA GRAMATICA

Sea una gramática $G=(\Sigma_T, \Sigma_N, S, P)$.

Se llama lenguaje asociado a la G, o **lenguaje generado por G**, o **lenguaje descrito por G, al conjunto:**

$$L(G) = \{ x / S \rightarrow^* x \text{ and } x \in \Sigma_T^* \}$$

Es el **conjunto** de todas las sentencias de la gramática

EQUIVALENCIA DE GRAMÁTICAS

G_1 y G_2 son equivalentes si generan el mismo lenguaje.

$G_1 \equiv G_2$, si $L(G_1) = L(G_2)$

Ejemplo:

$G_1 = (\{0,1\},\{A, B\}, A, P)$
Donde P:
A::= 1B1
A::= 0B0
B::= A
B::= 1
B::= 0
B::= λ

$G_2 = (\{0,1\},\{A, B, C, D\}, A, P)$
Donde P:
A::= 1B1
A::= 0C0
B::= D
C::= D
D::= A
D::= 1
D::= 0
D::= λ

G_1 y G_2 son equivalentes

REGLA COMPRESORA

La parte derecha está formada por menos símbolos que la parte izquierda.

Se denominan así porque transforman una palabra en otra de menor longitud.

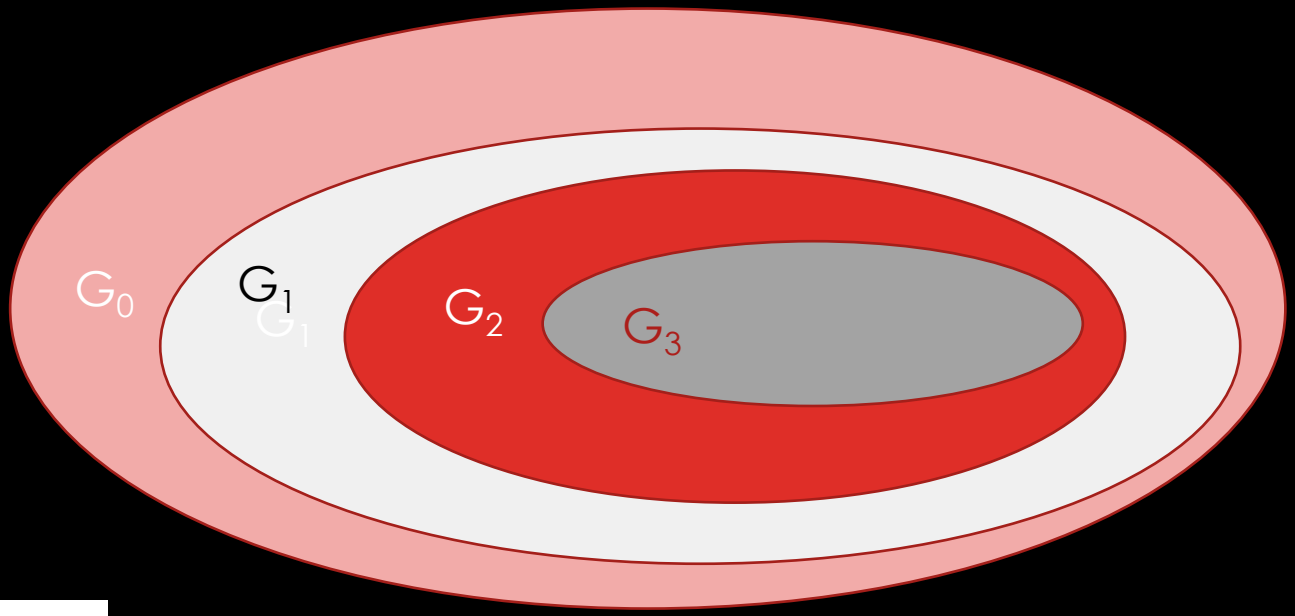
Son ejemplos las reglas del tipo **x ::= λ**

Ejemplo:

0C0 ::= A
A ::= λ

TIPOS DE GRAMÁTICAS. JERARQUÍA DE CHOMSKY

Noah Chomsky clasificó las gramáticas en cuatro grandes grupos: G_0, G_1, G_2 y G_3. Cada uno de estos grupos incluye las gramáticas del siguiente, de acuerdo con el siguiente esquema:



$$G3 \subset G2 \subset G1 \subset G0$$

Tipos de Gramáticas. Jerarquía de Chomsky

- **Gramáticas tipo 0**

"Sin –restricciones"

- Las reglas de producción tienen la forma

u ::= v

- En las reglas de producción:

✓ La parte izquierda no puede ser la palabra vacía.

✓ En la parte izquierda (u) debe aparecer algún símbolo no terminal

Ejemplo: sea G = ({a, b}, {A, B, C}, A, P), donde P:

A ::= aABC | abC
CB ::= BC
bB ::= bb
bC ::= b

✓**Define regla compresora**

. Gramáticas tipo 1

"Dependientes del contexto".

- Las reglas de producción de esta gramática tienen la forma

xAy ::= xvy

- *Se admite una excepción en la regla* **S ::= λ**

Ejemplo: G = ({S, B, C}, {a, b, b}, S,P), donde P es:

S ::= aSBc | aBC
AbB ::= abb
bC ::= bc
CB ::= BC
cC ::= cc
aB ::= ab

Tipos de Gramáticas. Jerarquía de Chomsky

- **Gramáticas tipo 2**

"Independiente de contexto"

•Las reglas de estas gramáticas se ajustan al siguiente esquema:

A ::= v

- *Se admite una excepción en la regla* **S ::= λ**

Ejemplo: sea la gramática G = ({a, b}, {S}, S, { S ::= aSb | ab})

Es una gramática de tipo 2. La derivación de la palabra aaabbb será:

S →aSb → aaSbb → aaabbb

Puede verse que el lenguaje definido por esta gramática es $\{a_n\ b_n \mid n=1, 2, ...\}$

. Gramáticas tipo 3

"Lenguajes Regulares"

- **Gramáticas lineales por la izquierda, cuyas reglas de producción pueden tener una de las formas siguientes:**

A ::= a

A ::= Va

S ::=λ

donde a $\in \Sigma_T$, A, V $\in \Sigma_N$, y S es el axioma de la gramática.

•**Gramáticas lineales por la derecha, cuyas reglas de producción tendrán la forma:**

A ::= a

A ::= aV

S ::= λ

donde a $\in \Sigma_T$, A, V $\in \Sigma_N$, y S es el axioma de la gramatica

TIPOS DE GRAMÁTICAS. JERARQUÍA DE CHOMSKY

Ejemplo de G3

- G1 = ({ 0, 1}, {A, B}, A, { A ::= B1 | 1, B ::= A0})

  Gramática lineal por la izquierda que describe el lenguaje:

  L1 = { 1, 101, 10101, ... } = {1(01)^n | n = 0, 1, 2, ...}

- G2 = ({ 0, 1}, {A, B}, A, { A ::= 1B | 1, B ::= 0A})

  Gramática lineal derecha que genera el mismo lenguaje que la gramática anterior.

ÁRBOLES DE DERIVACIÓN

- A toda derivación de una gramática de tipo 1, 2 ó 3 le corresponde un árbol de derivación. Este árbol se construye así:

Ejemplo: sea la gramática

G = ({0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0},{N,C}, N,

{N::= C | NC, C ::=0|1|2|3|4|5|6|7|8|9})

OBTENER LA PALABRA : 234

Consideraremos la derivación:

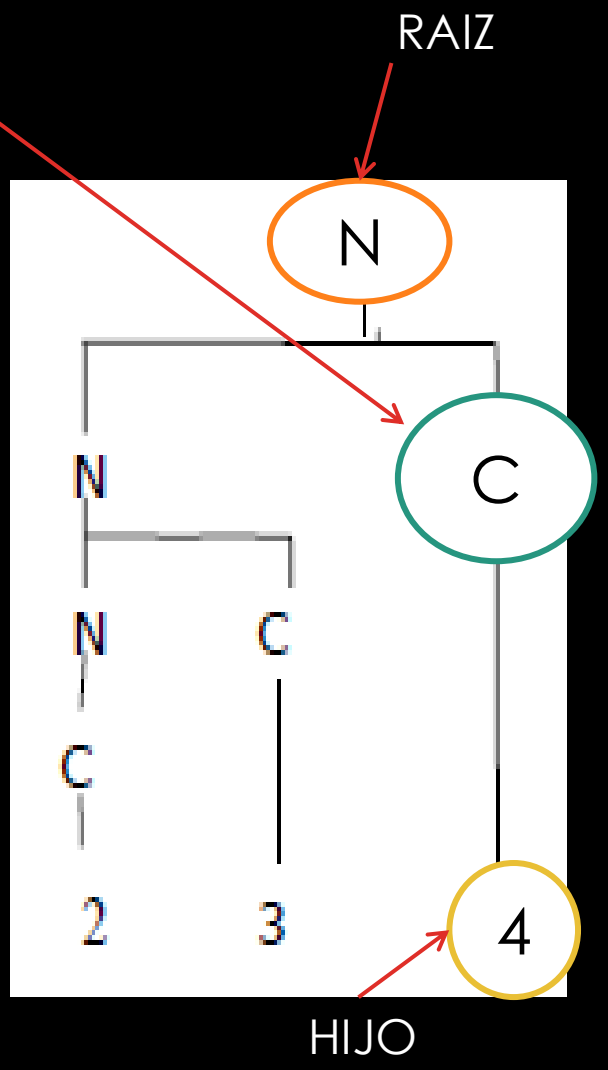
$$N \rightarrow NC \rightarrow NCC \rightarrow CCC \rightarrow 2CC \rightarrow 23C \rightarrow 234$$

ÁRBOLES DE DERIVACIÓN

Considerando la derivación:

N →NC → NCC → CCC → 2CC → 23C → 234

El árbol de derivación correspondiente es

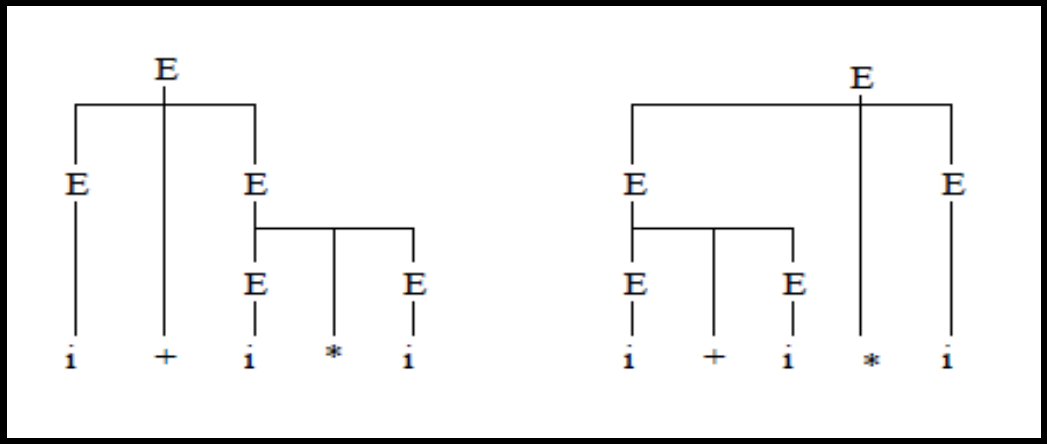


AMBIGÜEDAD

G = ({i, +, *, (,)}, {E}, E, E ::= E + E | E * E | (E) | i })

Consideremos la cadena **i+i*i.**

Para esta sentencia podemos tener los siguientes árboles de derivación:



Defina Ambigüedad:

AMBIGÜEDAD

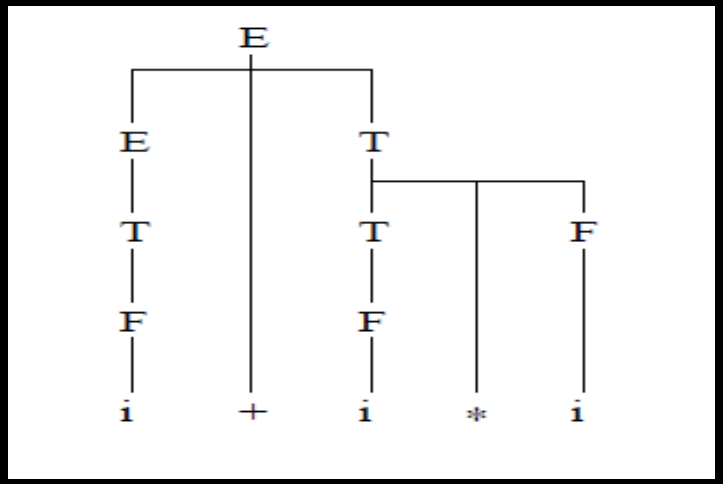
- Ejemplo: gramática equivalente a la anterior sin ser ambigua:

  La gramática anterior:

  G = ({i, +, *, (,)}, {E}, E, E ::= E + E | E * E | (E) | i })

  La gramática sin ambigüedad queda de la siguiente forma:

  G = ({i, +, *, (,)}, {E, T, F}, E, E ::= T | E + E, T ::= F | T * F, F ::= (E) | i })



RECURSIVIDAD

Una **gramática G se llama recursiva en A, A Є Σ_N, si**

$A \rightarrow^+ xAy$

- Se dice que **una producción es recursiva si A::= xAy**

- Si un lenguaje es infinito, la gramática que lo representa ha de ser recursiva

G = ({ i, +, *, (,) }, { E, T },E,

P={E ::= T+E | T*E | i ; T ::= E|(E) })

RECURSIVIDAD

Ejemplo:

- G recursiva

G = ({ i, +, *, (,) }, { E, T }, E, { E ::= T+E | T*E | i ; T ::= E|(E) })

Otra Notación

G =({ i, +, *, (,) }, { E, T }, E, P)

Donde P:

E ::= T+E

E ::= T*E

E ::= i

T::= E

T::= (E)

ELIMINACIÓN DE LA RECURSIVIDAD POR IZQUIERDA

Para cada A ε Σ_N

Si las producciones de A son:

P= (A ::= A . α_1 | A . α_2 |. . .| A . α_n | β_1 | β_2 |. . .| β_m)

donde β_i no comienza por **A**.

Entonces, se crea un *nuevo símbolo no terminal* **A'** :

$\Sigma_N = \Sigma_N$ U { **A'** };

P = (P – P) U { A ::= β_1 . A' | β_2 . A' |. . .| β_m . A' ,

A' = α_1 . A' | α_2 . A' |. . . | α_n . A' | λ }

ELIMINACIÓN DE LA RECURSIVIDAD POR IZQUIERDA

Ejemplo:

G =({ i, +, *, (,) }, { E, T }, E, P)

Donde P:

E ::= E + T

E ::= E * T

E ::= i

T::= E

T::= (E)

Sin cambios

Aplicando el Algoritmo nos queda:

E ::= i . E'

E' ::= + T . E'

E' ::= * T . E'

E' ::= λ

T::= E

T::= (E)

FACTORIZACIÓN A IZQUIERDA

- Es cuando en una gramática aparecen producciones de un mismo símbolo no terminal en cuya parte derecha, la primera parte es común.

- **Ejemplo: {E ::=E+E | E*E | i }.**

FACTORIZACIÓN A IZQUIERDA

Por cada $A \varepsilon \Sigma_N$

Si $A ::= \beta . \alpha_1 \mid \beta . \alpha_2$

Entonces cambiar esas producciones por:

$A ::= \beta . A'$

$A' ::= \alpha_1 \mid \alpha_2$

FACTORIZACIÓN A IZQUIERDA

- **Ejemplo:**

E ::= E + E
E ::= E * E
E ::= i.

Se crea E' y quedaría:

E ::= EE'
E'::= +E | *E
E ::= i

TRABAJO PRÁCTICO Nº 2

- Resolver los ejercicios del Trabajo Práctico Nº 2.

- Fecha Presentación: Publicada en Aula Virtual (Subir TP resuelto en el link "Entrega del Trabajo Práctico Nº 2").