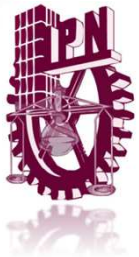
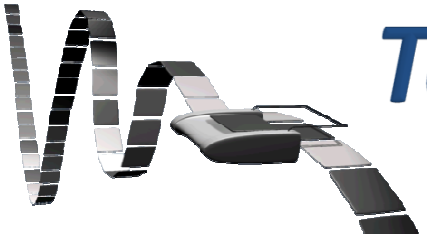


Instituto Politécnico Nacional

Escuela Superior de Cómputo



Teoría computacional

Clase 14: Gramáticas libres de contexto

M. en C. Edgardo Adrián Franco Martínez
http://computacion.cs.cinvestav.mx/~efranco
@efranco_escom
edfrancom@ipn.mx

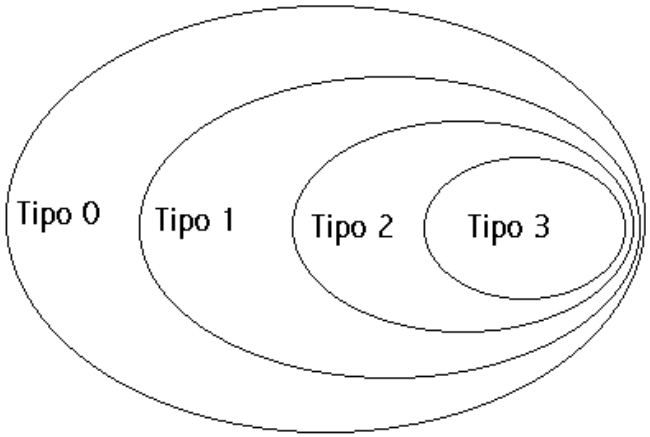
Contenido

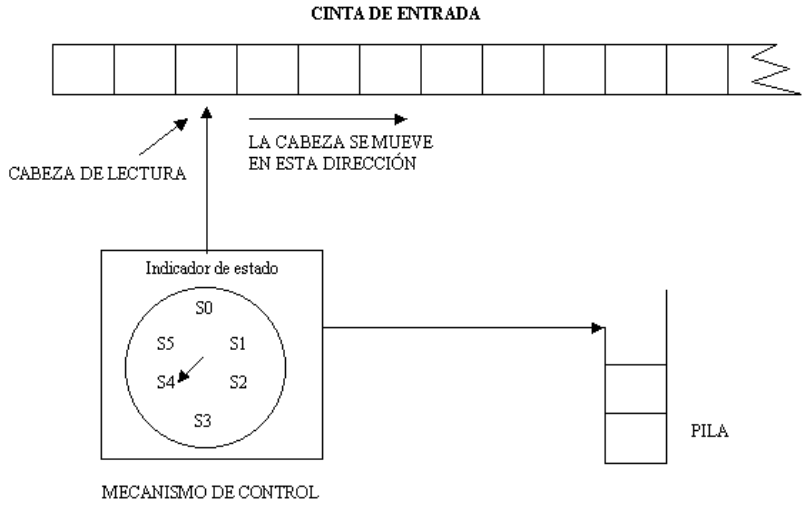
- Gramáticas libres de contexto
- Propiedades de los lenguajes libres de contexto
- Otros lenguajes libres de contexto
- BNF (Backus-Naur Form)
  - Ejemplo
- Árbol de derivación

Gramáticas libres de contexto

Las **gramáticas de tipo 2** o **gramáticas independientes del contexto**, son las que **generan los lenguajes libres o independientes del contexto**.

Los **lenguajes libres del contexto** son aquellos que **pueden ser reconocidos por un autómata de pila** determinístico o no determinístico.





- Una **gramática libre de contexto (GLC),** describe un **lenguaje libre de contexto**.

- Son útiles para describir **bloques anidados en lenguajes de programación** ya que **describen su sintaxis**.

- Son llamadas así porque el **elemento no terminal del lado derecho se puede sustituir sin importar el contexto en que este**.

- Su característica es que piden que solamente exista un **no terminal del lado izquierdo de la producción**.

$$A \rightarrow \alpha$$

siendo $A \in VN$ y $\alpha \in (VN \cup VT)^+$

Ejemplo 01

- Una simple gramática libre de contexto es

$$S \rightarrow aSb \mid \varepsilon$$

- Esta gramática genera el lenguaje no regular $\{a^n b^n : n \geq 0\}$

Ejemplo 02

- Gramática libre de contexto para expresiones enteras algebraicas sintácticamente correctas sobre las variables x, y y z:

$$S \rightarrow x \mid y \mid z \mid S + S \mid S - S \mid S * S \mid S/S \mid (S)$$

- cadena (x + y) *x - z *y / (x + x) valida

Ejemplo 03

- Lenguaje consistente en todas las cadenas que se pueden formar con las letras a y b, habiendo un número diferente de una que de otra, sería:

  $S \rightarrow U \mid V$

  $U \rightarrow TaU \mid TaT$

  $V \rightarrow TbV \mid TbT$

  $T \rightarrow aTbT \mid bTaT \mid \varepsilon$

  - *T genera todas las cadenas con la misma cantidad de letras a que b, U genera todas las cadenas con más letras a, y V todas las cadenas con más letras b.*

Ejemplo 04

- Gramática libre de contexto para el lenguaje

$$\{a^n b^m c^{m+n} : n \geq 0, m \geq 0\}$$

S → aSc | B

B → bBc | ε

Ejemplo 05

- La gramática **G = ({E,T,F}, {a,+,*,(,)}, S, P)**

  P:

  $E \rightarrow E + T \mid T$

  $T \rightarrow T * F \mid F$

  $F \rightarrow (E) \mid a$

Otros lenguajes libres de contexto

$$
\begin{aligned}
L_{ab} &= \{a^n b^n \mid n \geq 0\} \\
L_{abc} &= \{a^n b^n c^n \mid n \geq 0\} \\
L_{pal} &= \{w \mid w \in \{0,1\}^*, w = vv^R\} \\
L_{dup} &= \{w \mid w \in \{0,1\}^*, w = vv\} \\
L_{quad} &= \{0^{n^2} \mid n \text{ número cuadrado}\} \\
L_{prim} &= \{0^n \mid n \text{ número primo}\} \\
L_{()} &= \{w \mid w \in \{(,)\}^*, w \text{ correcto}\}
\end{aligned}
$$

BNF (Backus-Naur Form)

- Las gramáticas libres del contexto se escriben, frecuentemente, utilizando una notación conocida como BNF (Backus-Naur Form).
  - BNF es la técnica más común para definir la sintaxis de los lenguajes de programación.

Propiedades de los lenguajes libres de contexto

- El inverso de un lenguaje libre de contexto es también libre de contexto, pero el complemento no tiene por que serlo.

- La unión y concatenación de dos lenguajes libres de contexto es también libre de contexto. La intersección no tiene por que serlo.

- Los lenguajes regulares son libres de contexto porque pueden ser descritos mediante una gramática de libre contexto.

- La intersección de un lenguaje libre de contexto y un lenguaje regular es siempre libre de contexto.

- Para demostrar que un lenguaje dado no es libre de contexto, se puede emplear el Lema del bombeo para lenguajes libres de contexto.

- El problema de determinar si una gramática sensible al contexto describe un lenguaje libre del contexto es indecidible.

BNF (Ejemplo)

- La siguiente es una definición BNF del lenguaje que consiste de cadenas de paréntesis anidados:

<cadena_par>::=<cadena_par><paréntesis>|<paréntesis>

<paréntesis> ::= (<cadena_par>) |( )

  - Por ejemplo las cadenas ( ) (( )) y ( ) ( ) ( ) son cadenas válidas. En cambio las cadenas (() y ( ))) no pertenecen al lenguaje.

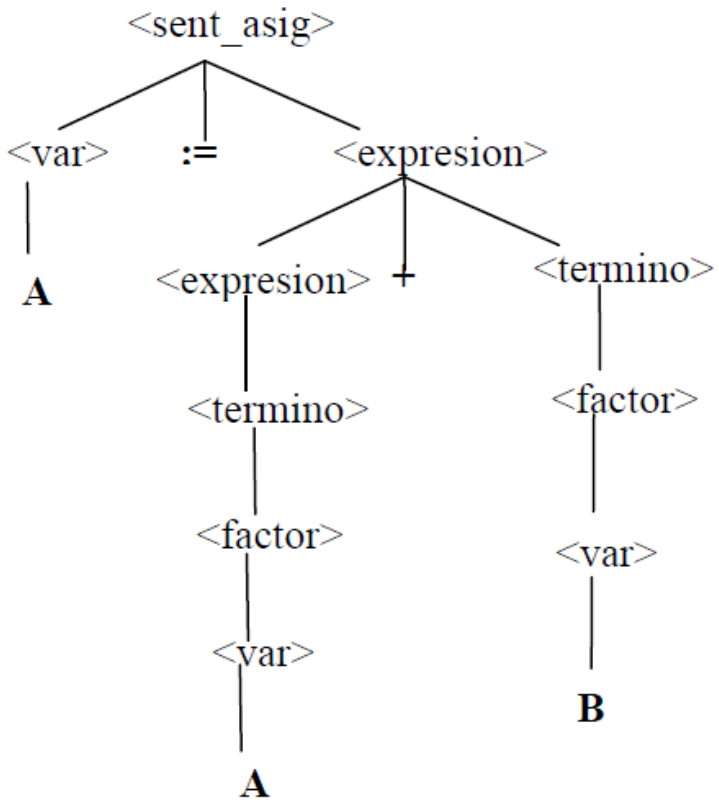
- En esta notación se deben seguir las siguientes convenciones:
  - Los no terminales se escriben entre paréntesis angulares <>
  - Los terminales se representan con cadenas de caracteres sin paréntesis angulares
  - El lado izquierdo de cada regla debe tener únicamente un no terminal *(ya que es una gramática libre del contexto)*
  - El símbolo ::=, que se lee “se define como” o “se reescribe como”, se utiliza en lugar de
  - Varias producciones del tipo
    - $\langle A \rangle ::= \langle B_1 \rangle$
    - $\langle A \rangle ::= \langle B_2 \rangle$ .. .
    - $\langle A \rangle ::= \langle B_n \rangle$
    - Se pueden escribir como $\langle A \rangle ::= \langle B_1 \rangle \mid \langle B_2 \rangle \mid \ldots \mid \langle B_n \rangle$

Árbol de derivación

La siguiente definición BNF describe la sintaxis (simplificada) de una sentencia de asignación de un lenguaje tipo Pascal:

<sent_asig> ::= <var> := <expresion>
<expresion> ::= <expresion> + <termino>| <expresion> - <termino> |<termino>
<termino> ::= <termino> * <factor>|<termino> / <factor>|<factor>
<factor> ::= (<expresion>)|<var> | <num>
<var> ::= A|B|C|D|... |Z
<num> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<sent_asig> ::= <var> := <expresion>
<expresion> ::= <expresion> + <termino>| <expresion> - <termino> |<termino>
<termino> ::= <termino> * <factor>|<termino> / <factor>|<factor>
<factor> ::= (<expresion>)|<var> | <num>
<var> ::= A|B|C|D|... |Z
<num> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9



A := A + B