Universidad Autónoma del Estado de Hidalgo

ICBI

Gramaticas libres de contexto

Jessica Ceron Perez

Semestre 6 Grupo 3

Dr. Eduardo Cornejo-Velázquez

Marco teórico

Las gramáticas libres de contexto y el formato Backus-Naur son conceptos fundamentales en la teoría de lenguajes formales y la informática. Estas herramientas permiten describir y analizar la estructura sintáctica de lenguajes.

GRAMATICAS LIBRES DE CONTEXTO
Una gramática libre de contexto se define como un 4-tupla (V, Σ, P, S), donde:

- V: conjunto de variables (no terminales)
- Σ : conjunto de símbolos terminales
- P: conjunto de producciones (reglas)
- S: símbolo inicial (no terminal)

Ejemplo:
Gramática simple para expresar números enteros:

- $V = \{N, D\}$
- $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
- $P = \{$
  - $N \rightarrow D$
  - $D \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$
  - $D \rightarrow DD$

$\}$
$S = N$
BACKUS-NAUR FORM (BNF)
El formato Backus-Naur es una notación utilizada para describir gramáticas libres de contexto.
Ejemplo: La gramática anterior se puede expresar en BNF como: `<N> ::= <D>`
`<D> ::= 0` | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |$< D >< D >$
`Características del BNF:`

- `Utiliza los símbolos` $<$ `y` $>$ `para delimitar no terminales.`
- `Utiliza` $::=$ `para indicar producciones.`
- `Utiliza` | `para separar alternativas.`
- `Utiliza espacios para separar símbolos.`

`Ventajas del BNF:`

- `Fácil de leer y escribir.`
- `Permite describir gramáticas complejas de manera concisa.`
- `Es independiente del lenguaje de programación.`

`Aplicaciones:`

- `Diseño de lenguajes de programación.`
- `Compiladores e intérpretes.`

- Análisis sintáctico.
- Procesamiento de lenguajes naturales.

Conclusión

Las gramáticas libres de contexto y el formato Backus-Naur son herramientas fundamentales para el diseño y análisis de lenguajes formales.

Referencias

1. Aho, A. V., Ullman, J. D. (1972). The theory of parsing, translation, and compiling. Prentice Hall.
2. Hopcroft, J. E., Ullman, J. D. (1979). Introduction to automata theory, languages, and computation. Addison-Wesley.

GRAMATICAS LIBRES DEL CONTEXTO

Estas gramáticas, conocidas también como 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. Como toda gramática se definen mediante una cuadrupla G = (N, T, P, S), siendo:

- N es un conjunto finito de símbolos no terminales
- T es un conjunto finito de símbolos terminales

$$N \cap T = \emptyset$$

- P es un conjunto finito de producciones
- S es el símbolo distinguido o axioma

$$S \notin (N \cup T)$$

En una gramática libre del contexto, cada producción de P tiene la forma:

$$A \rightarrow \omega \qquad \begin{cases} A \in N \cup \{S\} \\ \omega \in (N \cup T)^* - \{\varepsilon\} \end{cases}$$

Es decir, que en el lado izquierdo de una producción pueden aparecer el símbolo distinguido o un símbolo no terminal y en el lado derecho de una producción cualquier cadena de símbolos terminales y/o no terminales de longitud mayor o igual que 1.
La gramática puede contener también la producción

$$S \rightarrow \notin$$

si el lenguaje que se quiere generar contiene la cadena vacía.
Ejemplo:
La siguiente gramática genera las cadenas del lenguaje $L1 = \{wcw^R \mid w \in \{a,b\}^*\}$:

$$G1 = (\{A\}, \{a,b,c\}, P1, S1)$$

donde $P1$ contiene las siguientes producciones:

$$S1 \rightarrow A$$

$$A \rightarrow aAa$$

$$A \rightarrow bAb$$

$$A \rightarrow c$$

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. 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: <A> ::= <B1> $|< B2 >| \ldots |< Bn >$ Ejemplo:
La siguiente es una definición BNF del lenguaje que consiste de cadenas de paréntesis anidados:
Por ejemplo, las cadenas ( ), (( )) y ( ) ( ) ( ) son cadenas válidas.
En cambio, las cadenas ((y ( ))) no pertenecen al lenguaje.

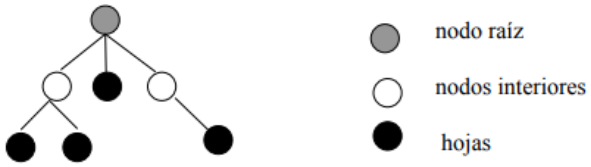
Arbol de derivación

Un árbol de derivación permite mostrar gráficamente cómo se puede derivar cualquier cadena de un lenguaje a partir del símbolo distinguido de una gramática que genera ese lenguaje. Un árbol es un conjunto de puntos, llamados nodos, unidos por líneas, llamadas arcos. Un arco conecta dos nodos distintos. Para ser un árbol un conjunto de nodos y arcos debe satisfacer ciertas propiedades:

- hay un único nodo distinguido, llamado raíz (se dibuja en la parte superior) que no tiene arcos incidentes
- todo nodo c excepto el nodo raíz está conectado con un arco a otro nodo k, llamado el padre de c (c es el hijo de k). El padre de un nodo, se dibuja por encima del nodo.
- todos los nodos están conectados al nodo raíz mediante un único camino.
- los nodos que no tienen hijos se denominan hojas, el resto de los nodos se denominan nodos interiores.

Popiedades del árbol de derivación:

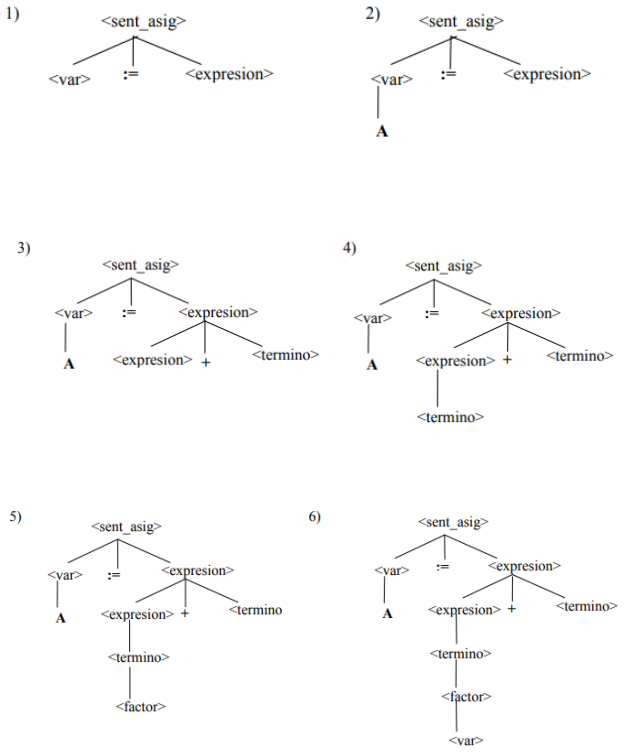
1. el nodo raíz está rotulado con el símbolo distinguido de la gramática
2. cada hoja corresponde a un símbolo terminal o un símbolo no terminal
3. cada nodo interior corresponde a un símbolo no terminal



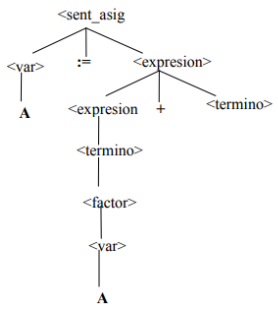
Para cada cadena del lenguaje generado por una gramática es posible construir (al menos) un árbol de derivación, en el cual cada hoja tiene como rótulo uno de los símbolos de la cadena.
Ejemplo:
La siguiente definición BNF describe la sintaxis (simplificada) de una sentencia de asignación de un lenguaje 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
```

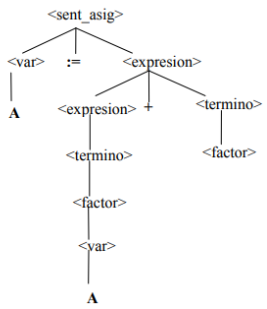
Por ejemplo, la sentencia A := A + B es una sentencia de asignación que pertenece al lenguaje definido por la definición BNF dada, y cuyo árbol de derivación se construye como se muestra a continuación:



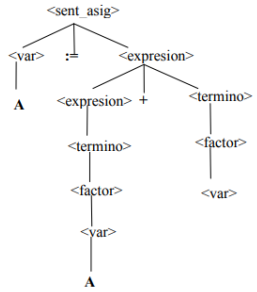
7)



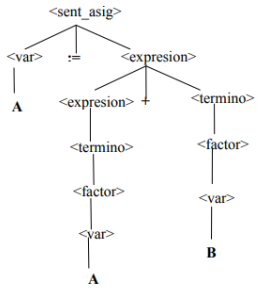
8)



9)



10)



Referencias Bibliográficas

References

[1] Gramáticas, E., Del contexto, C. T. C. G. de T. 2. o. G. I., de pila determinístico o no determinístico., La forma, C. P. de P. T. (s/f) *Gramáticas libres del contexto. Edu.ar.*, https://users.exa.unicen.edu.ar/catedras/ccomp1/Apunte5.pdf