Computabilidad y Complejidad (CMC)

Práctica 1: Gramáticas formales generativas

José M. Sempere

Valencian Research Institute for Artificial Intelligence (VRAIN)
Valencian Graduate School and Research Network of Artificial Intelligence (valgrAI)
Department of Informatics Systems and Computation (DSIC)
Universitat Politècnica de València (UPV)

http:// jsempere.webs.upv.es

jsempere@dsic.upv.es

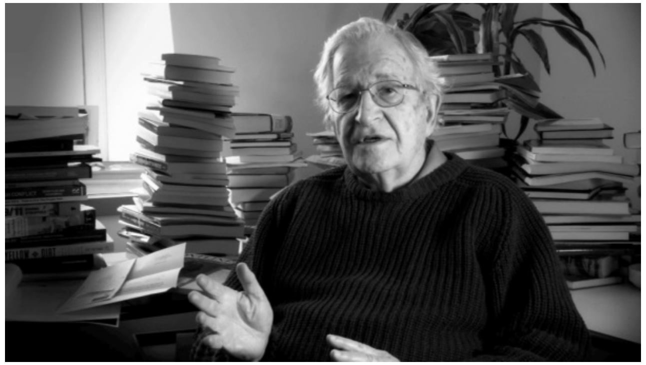
Índice

1. Definición de gramática. Lenguaje generado por una gramática
2. Implementación en Mathematica
3. La Jerarquía de Chomsky
4. Análisis de cadenas en gramáticas incontextuales: El algoritmo CYK
5. Actividades propuestas

Bibliografía básica recomendada

- Introduction to Automata Theory, Languages and Computation (J.E. Hopcroft, R. Motwani, J.D. Ullman - Addison Wesley, 2001)
- Teoría de la computación (J. Glenn Brookshear - Addison Wesley Iberoamericana, 1993)

Gramáticas generativas



Noam Chomsky

Según la RAE, la gramática es *“la ciencia que estudia los elementos de una lengua y sus combinaciones.”* y *“el tratado de esta ciencia*”. En nuestro caso, nos interesan las gramáticas desde el punto de vista de los lenguajes formales. Es decir que nos interesan las gramáticas conocidas como gramáticas generativas que según la RAE son aquellas “ … *que tratan de formular una serie de reglas capaces de generar o producir todas las oraciones posibles y aceptables de un idioma*” (en nuestro caso, de un lenguaje formal. Nos referiremos a las gramáticas generativas para los lenguajes formales como gramáticas formales.

Los modelos de gramáticas generativas, en sus ideas básicas, siguen fundamentalmente la “teoría estándar” propuesta por Noam Chomsky hacia 1965 [1] (basándose en su propia tesis doctoral y trabajos anteriores como en [2]).

En el caso de las gramáticas formales, la formulación de este modelo permite establecer de forma rigurosa qué sentencias están correctamente formuladas en un lenguaje y cuáles no. Más aún, la decisión acerca de la corrección o no de una sentencia se puede realizar de forma automática y eficiente en algunos casos, lo que ha dado lugar al diseño efectivo de compiladores e intérpretes en el marco de la Informática. Por ejemplo, en la actualidad operamos con compiladores e intérpretes para lenguajes de programación, lenguajes de especificación, lenguajes de control de procesos, etc.

Bibliografía adicional

[1] N. Chomsky. *Aspects of the theory of syntax*. Cambridge, Massachusetts: MIT Press. 1965
[2] N. Chomsky. *Syntactic Structures*. The Hague: Mouton.1957

Definición formal de las gramáticas generativas

$$G = (N, T, P, S)$$

- N y T son alfabetos de símbolos auxiliares y terminales respectivamente con $N \cap T = \emptyset$
- $S \in N$ es el símbolo inicial o axioma
- P es un conjunto finito de producciones. Una producción la definiremos como un par (α, β) donde
  - α es la parte izquierda ó antecedente de la producción y es una cadena sobre $(N \cup T)^* N (N \cup T)^*$
  - β es la parte derecha ó consecuente de la producción y es una cadena sobre $(N \cup T)^*$

La producción (α, β) la escribiremos como $\alpha \rightarrow \beta$ (leído " … *alfa produce beta*")
Si nos encontramos con un conjunto de producciones que comparten antecedente $(\alpha, \beta_1), (\alpha, \beta_2), \ldots, (\alpha, \beta_n)$
entonces escribiremos $\alpha \rightarrow \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n$

Notación

Habitualmente escribiremos las gramáticas de forma compacta, especificando únicamente sus producciones y siguiendo el convenio establecido a continuación :

(a) los símbolos del abecedario en mayúscula se referirán a símbolos auxiliares
(b) los símbolos del abecedario en minúscula, los dígitos y cualquier otro símbolo especial se referirán a símbolos terminales
(c) el axioma se escribirá como S

Cualquier alteración del anterior convenio se establecerá de forma explícita.

Algunos ejemplos

Ejemplo 1

$$S \rightarrow aBSc \mid abc$$
$$Ba \rightarrow aB$$
$$Bb \rightarrow bb$$

Ejemplo 2

$$S \rightarrow U \mid V$$
$$U \rightarrow TaU \mid TaT$$
$$V \rightarrow TbV \mid TbT$$
$$T \rightarrow aTbT \mid bTaT \mid \lambda$$

Ejemplo 3

$$S \rightarrow A$$
$$A \rightarrow aA \mid B$$
$$B \rightarrow bB \mid C$$
$$C \rightarrow cC \mid \lambda$$

Lenguaje generado por una gramática

Relación de derivación directa

Sea $G = (N, T, P, S)$ una gramática y α, β dos cadenas sobre $(N \cup T)^*$. Diremos que α deriva directamente en β de acuerdo con G, denotándolo por $\alpha \underset{G}{\Rightarrow} \beta$ si se cumple que $\alpha = \alpha_1\alpha_2\alpha_3$, $\beta = \alpha_1\beta_1\alpha_3$ y $\alpha_2 \rightarrow \beta_1 \in P$

Relación de derivación

Sea $G = (N, T, P, S)$ una gramática y α, β dos cadenas sobre $(N \cup T)^*$

Diremos que α deriva en β de acuerdo con G, denotándolo por $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$ si se cumple una de las dos siguientes condiciones:

(a) $\alpha = \beta$

(b) existe una cadena γ de forma que $\alpha \underset{G}{\Rightarrow} \gamma \overset{*}{\underset{G}{\Rightarrow}} \beta$

Formas sentenciales, palabras y lenguajes

Sea $G = (N, T, P, S)$ una gramática. Diremos que α es …

(a) una forma sentencial de G si $\alpha \in (N \cup T)^*$ y $S \overset{*}{\underset{G}{\Rightarrow}} \alpha$

(b) una palabra o cadena de G si $\alpha \in T^*$ y $S \overset{*}{\underset{G}{\Rightarrow}} \alpha$

Lenguaje generado por una gramática G

Sea $G = (N, T, P, S)$ una gramática. El lenguaje generado por G se define como el conjunto

$$L(G) = \{x \in T^* : S \overset{*}{\underset{G}{\Rightarrow}} x\}$$

Equivalencia entre gramáticas

Dos gramáticas G_1 y G_2 son equivalentes si $L(G_1) = L(G_2)$

Notación: Los símbolos $\underset{G}{\Rightarrow}$ y $\overset{*}{\underset{G}{\Rightarrow}}$ los reescribiremos como $\Rightarrow$ y $\overset{*}{\Rightarrow}$ siempre que la gramática G quede sobreentendida

Algunos ejemplos

Ejemplo 1

$$S \rightarrow aBSc \mid abc$$
$$Ba \rightarrow aB$$
$$Bb \rightarrow bb$$

Genera el lenguaje de cadenas $\{a^n b^n c^n : n \geq 1\}$

Ejemplo 2

$$S \rightarrow U \mid V$$
$$U \rightarrow TaU \mid TaT$$
$$V \rightarrow TbV \mid TbT$$
$$T \rightarrow aTbT \mid bTaT \mid \lambda$$

Genera el lenguaje de cadenas formadas por símbolos a y b donde el número de símbolos a es distinto del número de símbolos b

Ejemplo 3

$$S \rightarrow A$$
$$A \rightarrow aA \mid B$$
$$B \rightarrow bB \mid C$$
$$C \rightarrow cC \mid \lambda$$

Genera el lenguaje de cadenas pertenecientes a la expresión regular $a^* b^* c^*$

Codificación de las gramáticas en *Mathematica*

- Las cadenas en *Mathematica* se codifican mediante listas. Por ejemplo, la cadena $abbba$ se representa por la lista $\{a, b, b, b, a\}$ y la cadena vacía λ se representa como la lista vacía $\{ \}$

- La representación de la gramática $G = (N, T, P, S)$ en *Mathematica*, la haremos mediante una lista de cuatro elementos: la lista N, la lista T, la lista P y el elemento S

$$G = \{N, T, P, S\}$$

- P será una lista de listas, donde por cada antecedente común de las producciones, tendremos una lista.

Ejemplo 1

$$S \to aBSc \mid abc$$
$$Ba \to aB$$
$$Bb \to bb$$

$$G = \{ \{S, B\}, \{a, b, c\}, P, S\}$$

P = { producciones_de_S, producciones_de_Ba, producciones_de_Bb}

Codificación de las gramáticas en *Mathematica*

Ejemplo 1

$S \to aBSc \mid abc$

$Ba \to aB$

$Bb \to bb$

$G = \{\{S, B\}, \{a, b, c\}, P, S\}$

P={ producciones_de_S, producciones_de_Ba, producciones_de_Bb}

Las producciones de cada antecedente común se representarán mediante una lista de dos listas (una lista para la parte izquierda común y una lista con las partes derechas)

$$S \to aBSc \mid abc$$

producción $S \to aBSc$

producción $S \to abc$

producciones_de_S={{S},{{a,B,S,c},{a,b,c}}}

parte izquierda común
(una lista con el antecedente)

partes derechas de las producciones
(una lista con tantas listas como partes derechas existan)

Algunos ejemplos

Ejemplo 1

$$S \to aBSc \mid abc$$
$$Ba \to aB$$
$$Bb \to bb$$

{{S,B},{a,b,c},{{{S},{{a,B,S,c},{a,b,c}}},{{B,a},{{a,B}}},{{B,b},{{b,b}}}},S}

Ejemplo 2

$$S \to U \mid V$$
$$U \to TaU \mid TaT$$
$$V \to TbV \mid TbT$$
$$T \to aTbT \mid bTaT \mid \lambda$$

{{S,U,V,T},{a,b},{{{S},{{U},{V}}},{{U},{{T,a,U},{T,a,T}}},{{V},{{T,b,V},{T,b,T}}},{{T},{{a,T,b,T},{b,T,a,T},{}}}},S}

Ejemplo 3

$$S \to A$$
$$A \to aA \mid B$$
$$B \to bB \mid C$$
$$C \to cC \mid \lambda$$

{{S,A,B,C},{a,b,c},{{{S},{{A}}},{{A},{{a,A},{B}}},{{B},{{b,B},{C}}},{{C},{{c,C},{}}}},S}

La Jerarquía de Chomsky

La Jerarquía de Chomsky es un marco de estudio de los lenguajes formales basado en una taxonomía de las gramáticas en función de la forma que toman las producciones. Se proponen cuatro grandes familias de gramáticas que, a su vez, definen cuatro familias de lenguajes.

Gramáticas de tipo 0 ó no restringidas

No tienen ningún tipo de restricción.

Gramaticas de tipo 1 ó sensibles al contexto

Las producciones toman la forma

$$\alpha A\beta \to \alpha\gamma\beta,\ \ \alpha,\beta \in\ (N \cup T)^*,\ \ A \in N,\ \ \gamma \in (N \cup T)^+$$

Adicionalmente se permite la regla $S \to \lambda$ siempre que S no aparezca como consecuente en ninguna producción.

Gramáticas de tipo 2 ó de contexto libre ó incontextuales

Las producciones toman la forma

$$A \to \alpha \quad A \in N,\ \ \alpha \in\ (N \cup T)^*$$

Gramáticas de tipo 3 ó regulares

Gramáticas lineales por la derecha

Las producciones toman la forma

$$A \to aB \quad a \in T, \qquad A, B \in N$$

$$A \to a \quad a \in T, \qquad A \in N$$

$$A \to B \quad A, B \in N$$

$$A \to \lambda \quad A \in N$$

Gramáticas lineales por la izquierda

Las producciones toman la forma

$$A \to Ba \quad a \in T, \qquad A, B \in N$$

$$A \to a \quad a \in T, \qquad A \in N$$

$$A \to B \quad A, B \in N$$

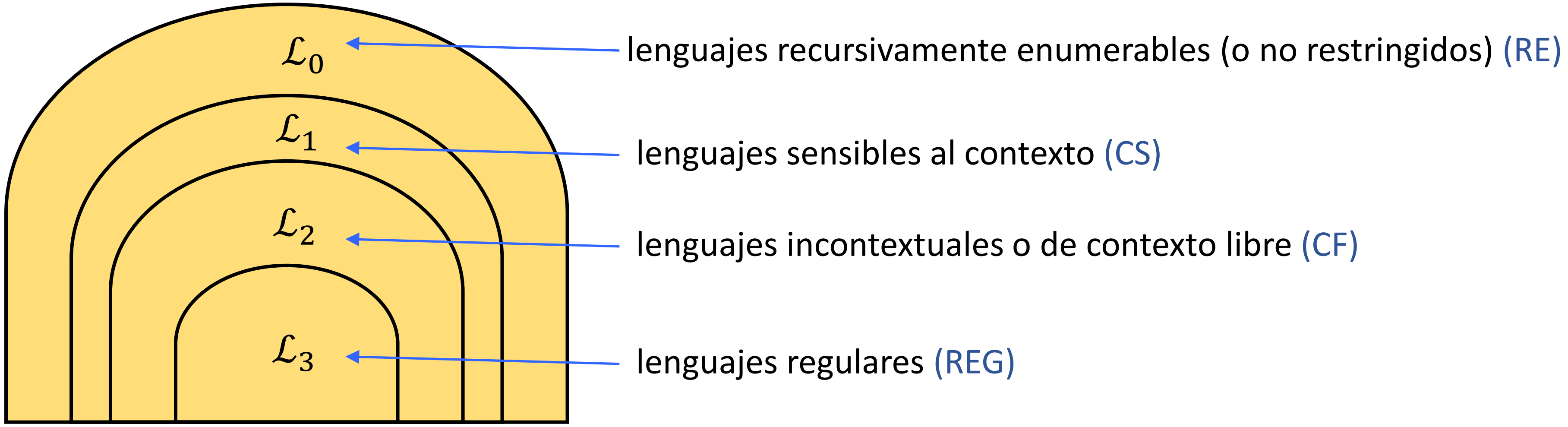
$$A \to \lambda \quad A \in N$$

La Jerarquía de Chomsky

Desde el punto de vista de las clases de lenguajes, los cuatro tipos de gramáticas definidas en la jerarquía de Chomsky definen cuatro familias de lenguajes, atendiendo al siguiente criterio:

"Un lenguaje es de tipo i si existe una gramática de tipo i que lo genera"

De esta forma y desde el punto de vista de las clases de lenguajes que originan los cuatro tipos de gramáticas, la jerarquía de Chomsky establece una relación de clases de lenguajes que obedece al siguiente esquema



Algunos ejemplos

Ejemplo 1

$$S \to aBSc \mid abc$$
$$Ba \to aB$$
$$Bb \to bb$$

Gramática de tipo 0 (existe una gramática equivalente de tipo 1)

Ejemplo 2

$$S \to U \mid V$$
$$U \to TaU \mid TaT$$
$$V \to TbV \mid TbT$$
$$T \to aTbT \mid bTaT \mid \lambda$$

Gramática de tipo 2

Ejemplo 3

$$S \to A$$
$$A \to aA \mid B$$
$$B \to bB \mid C$$
$$C \to cC \mid \lambda$$

Gramática de tipo 3 lineal por la derecha

Análisis de cadenas en gramáticas incontextuales

$$G = (N, T, P, S)$$

Consideramos que las gramáticas incontextuales bajo estudio están simplificadas. Esto significa que:

(1) Cada símbolo A es útil, es decir, participa en alguna derivación como la siguiente

$$S \overset{*}{\Rightarrow} \alpha A \beta \overset{*}{\Rightarrow} w \quad \begin{array}{l} A \in (N \cup T), \\ \alpha, \beta \in (N \cup T)^*, \\ w \in T^* \end{array}$$

(2) No existen producciones unitarias $A \rightarrow B \quad A, B \in N$

(3) No existen producciones vacías $A \rightarrow \lambda \quad A \in N$

Además, consideraremos que están en Forma Normal de Chomsky, lo que significa que sus producciones están en una de las dos siguientes formas:

- $A \rightarrow BC \quad A, B, C \in N$
- $A \rightarrow a \quad A \in N, \quad a \in T$

Algoritmo de análisis de cadenas en gramáticas incontextuales CYK (Cocke-Younger-Kasami)

Entrada : $G = (N, T, P, S)$ en Forma Normal de Chomsky y $w = w_1 w_2 \ldots w_n$ $(w \neq \lambda)$

Salida : *Cierto* (si $w \in L(G)$) o *Falso* (si $w \notin L(G)$)

Método :

Para i=1 hasta n
    $V_{i1} = \{ A : A \rightarrow w_i \in P \}$
finPara
Para j=2 hasta n
    Para i=1 hasta n-j+1
        $V_{ij} = \varnothing$
        Para k=1 hasta j-1
            $V_{ij} = V_{ij} \cup \{ A : A \rightarrow BC \in P, B \in V_{ik}, C \in V_{i+k,\, j-k} \}$
        finPara
    finPara
finPara
Si $S \in V_{1n}$ devolver *Cierto*
sino devolver *Falso*

- V_{ij} contiene todos aquellos auxiliares que pueden derivar la cadena w_{ij} en G.
- La cadena w_{ij} es la subcadena de w que comienza en la posición i y tiene longitud j

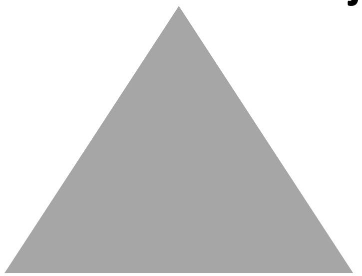
Algoritmo de análisis de cadenas en gramáticas incontextuales CYK (Cocke-Younger-Kasami)

Los conjuntos V_{ij}

posición i

longitud j

$\mathbf{A} \in \mathbf{V_{ij}}$



$$w = w_1 w_2 w_3 \dots \boxed{w_i \dots w_{i+j-1}} \dots w_{n-1} w_n$$

$$A \overset{*}{\Rightarrow} w_i \dots w_{i+j-1}$$

Algoritmo de análisis de cadenas en gramáticas incontextuales CYK (Cocke-Younger-Kasami)

Entrada : $G = (N, T, P, S)$ en Forma Normal de Chomsky y $w = w_1 w_2 \dots w_n$ ($w \neq \lambda$)

Salida : *Cierto* (si $w \in L(G)$) o *Falso* (si $w \notin L(G)$)

Método :

```
Para i=1 hasta n
        V_i1 = { A : A → w_i ∈ P }
finPara
Para j=2 hasta n
    Para i=1 hasta n-j+1
        V_ij = ∅
        Para k=1 hasta j-1
            V_ij = V_ij ∪ { A : A → BC ∈ P, B ∈ V_ik, C ∈ V_i+k, j-k }
        finPara
    finPara
finPara
Si S ∈ V_1n devolver Cierto
sino devolver Falso
```

Algoritmo de análisis de cadenas en gramáticas incontextuales CYK (Cocke-Younger-Kasami)

- Para i=1 hasta n …

$A \rightarrow w_1 \in P$ $\quad$ $A \rightarrow w_2 \in P$ $\quad$ $A \rightarrow w_n \in P$

$\mathbf{A \in V_{11}}$ $\quad$ $\mathbf{A \in V_{21}}$ ------------------------- $\mathbf{A \in V_{n1}}$

$\boxed{w_1} w_2 \; ... \; w_{n-1} w_n$ $\quad$ $w_1 \boxed{w_2} \; ... \; w_{n-1} w_n$ $\quad$ $w_1 \; w_2 \; ... \; w_{n-1} \boxed{w_n}$

Algoritmo de análisis de cadenas en gramáticas incontextuales CYK (Cocke-Younger-Kasami)

Entrada : $G = (N, T, P, S)$ en Forma Normal de Chomsky y $w = w_1 w_2 \ldots w_n$ $(w \neq \lambda)$

Salida : *Cierto* (si $w \in L(G)$) o *Falso* (si $w \notin L(G)$)

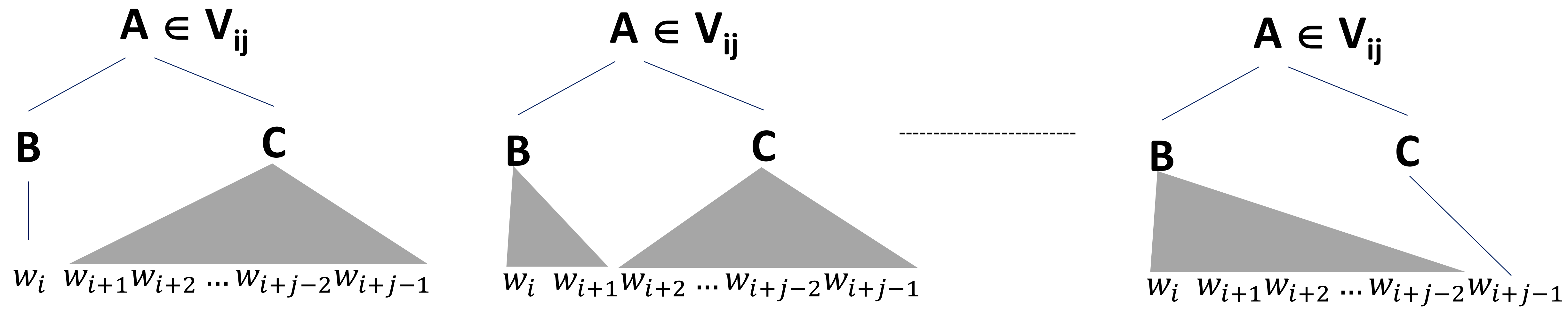
Método :

Para i=1 hasta n
    $V_{i1} = \{ A : A \to w_i \in P \}$
finPara
Para j=2 hasta n
    Para i=1 hasta n-j+1
        $V_{ij} = \varnothing$
        Para k=1 hasta j-1
            $V_{ij} = V_{ij} \cup \{ A : A \to BC \in P, B \in V_{ik}, C \in V_{i+k, j-k} \}$
        finPara
    finPara
finPara
Si $S \in V_{1n}$ devolver *Cierto*
sino devolver *Falso*

Algoritmo de análisis de cadenas en gramáticas incontextuales CYK (Cocke-Younger-Kasami)

- Para k=1 hasta j-1 …

***k* = 1** ***k* = 2** ***k* = j-1**



$A \rightarrow BC \in P$

$B \in V_{i1}$
$C \in V_{i+1,j-1}$

$A \rightarrow BC \in P$

$B \in V_{i2}$
$C \in V_{i+2,j-2}$

$A \rightarrow BC \in P$

$B \in V_{ij-1}$
$C \in V_{i+j-1,1}$

Algoritmo de análisis de cadenas en gramáticas incontextuales CYK (Cocke-Younger-Kasami)

$w_1\, w_2 \;\ldots\; w_{k-1} w_k$

posición

i →

longitud

j ↓

	1	2	…	…	k
1	V_{11}	V_{21}	…	…	V_{k1}
2	V_{12}	V_{22}	…	…	
	…	…	…		
	…	…			
k	V_{1k}				

Algoritmo de análisis de cadenas en gramáticas incontextuales CYK (Cocke-Younger-Kasami)

Ejemplo de aplicación del algoritmo CYK

$G = (N, T, P, S)$ (en Forma Normal de Chomsky)

S → AB | BC
A → BA | a
B → CC | b
C → AB | a

$w = baaba$

i →
j ↓

	b 1	a 2	a 3	b 4	a 5
1	B	A,C	A,C	B	A,C
2	S,A	B	S,C	S,A	
3	∅	B	B		
4	∅	V_{24}			
5					

Algoritmo de análisis de cadenas en gramáticas incontextuales CYK (Cocke-Younger-Kasami)

Ejemplo de aplicación del algoritmo CYK

$G = (N, T, P, S)$ (en Forma Normal de Chomsky)

$S \to AB \mid BC$
$A \to BA \mid a$
$B \to CC \mid b$
$C \to AB \mid a$

$w = baaba$

i →, j ↓

	b 1	a 2	a 3	b 4	a 5
1	B	A,C	A,C	B	A,C
2	S,A	B	S,C	S,A	
3	$\varnothing$	B	B		
4	$\varnothing$	{S,C}			
5					

Cálculo de V_{24}

- $k = 1$

$X : X \to YZ,\ Y \in V_{21},\ Z \in V_{33}$

$X : X \to YZ,\ Y \in \{A,C\},\ Z \in \{B\}$

$X : X \to AB \mid CB$

$S \in V_{24} \quad S \to AB$
$C \in V_{24} \quad C \to AB$

Algoritmo de análisis de cadenas en gramáticas incontextuales CYK (Cocke-Younger-Kasami)

Ejemplo de aplicación del algoritmo CYK

$G = (N, T, P, S)$ (en Forma Normal de Chomsky)

S → AB | BC
A → BA | a
B → CC | b
C → AB | a

$w = baaba$

i →, j ↓

	b 1	a 2	a 3	b 4	a 5
1	B	A,C	A,C	B	A,C
2	S,A	B	S,C	S,A	
3	∅	B	B		
4	∅	{S,C,A}			
5					

Cálculo de V_{24}

- $k = 2$

X : X → YZ, Y ∈ V_{22} , Z ∈ V_{42}

X : X → YZ, Y ∈ {B} , Z ∈ {S,A}

X : X → BS | BA A ∈ V_{24} A → BA

Algoritmo de análisis de cadenas en gramáticas incontextuales CYK (Cocke-Younger-Kasami)

Ejemplo de aplicación del algoritmo CYK

$G = (N, T, P, S)$ (en Forma Normal de Chomsky)

$S \rightarrow AB \mid BC$
$A \rightarrow BA \mid a$
$B \rightarrow CC \mid b$
$C \rightarrow AB \mid a$

$w = baaba$

i →, j ↓

j \ i	1 (b)	2 (a)	3 (a)	4 (b)	5 (a)
1	B	A,C	A,C	B	A,C
2	S,A	B	S,C	S,A	
3	∅	B	B		
4	∅	{S,C,A}			
5					

Cálculo de V_{24}

- $k = 3$

$X : X \rightarrow YZ,\ \ Y \in V_{23}\ ,\ Z \in V_{51}$

$X : X \rightarrow YZ,\ \ Y \in \{B\}\ ,\ Z \in \{A,C\}$

$X : X \rightarrow BA \mid BC$ $\qquad A \in V_{24} \qquad A \rightarrow BA$

$\qquad S \in V_{24} \qquad S \rightarrow BC$

Algoritmo de análisis de cadenas en gramáticas incontextuales CYK (Cocke-Younger-Kasami)

Ejemplo de aplicación del algoritmo CYK

$G = (N, T, P, S)$ (en Forma Normal de Chomsky)

S → AB | BC
A → BA | a
B → CC | b
C → AB | a

$w = baaba$

i →

j ↓

	1 (b)	2 (a)	3 (a)	4 (b)	5 (a)
1	B	A,C	A,C	B	A,C
2	S,A	B	S,C	S,A	
3	∅	B	B		
4	∅	S,C,A			
5	S,A,C				

$S \in V_{15}$

Cierto ($w \in L(G)$)

Actividades propuestas

1. Dada una gramática incontextual, diremos que un símbolo auxiliar es *directamente generativo* si aparece como antecedente de una producción cuyo consecuente es una cadena de símbolos terminales (incluida la cadena vacía). Implemente un módulo *Mathematica* que, tomando como entrada una gramática incontextual, obtenga como salida una lista con aquellos símbolos auxiliares de la gramática *directamente generativos*.

2. Dada una gramática incontextual, diremos que un símbolo auxiliar es *directamente no generativo* si en las producciones donde aparece como antecedente aparece también en el consecuente. Implemente un módulo *Mathematica* que, tomando como entrada una gramática incontextual, obtenga como salida una lista con aquellos símbolos auxiliares de la gramática *directamente no generativos*.

3. Dada una gramática incontextual en Forma Normal de Chomsky G y una cadena $w \neq \lambda$, implemente un módulo *Mathematica* que devuelva True si w pertenece a $L(G)$ y False en caso contrario.