Algoritmo CKY

Clase 19

IIC 2223

Prof. Cristian Riveros

Clase anterior: Bug en la demostración

Sea $L \subseteq \Sigma^*$. Si L es **libre de contexto** entonces:

$$(\mathrm{LB}^{\mathrm{CFL}}) \quad \begin{array}{l} \text{existe un } N > 0 \text{ tal que} \\ \text{para toda palabra} \quad z \in L \quad \text{con} \quad |z| \geq N \\ \text{existe una descomposición} \quad z = u\, v\, w\, x\, y \\ \qquad\quad \text{con} \quad vx \neq \epsilon \quad \text{y} \quad |vwx| \leq N \quad \text{tal que} \\ \text{para todo } i \geq 0, \quad u \cdot v^i \cdot w \cdot x^i \cdot y \in L. \end{array}$$

Demostración

(PIZARRA)

Consecuencias: unión, intersección y complemento

Proposición

Para todo lenguajes libres de contexto L_1 y L_2,
$L_1 \cup L_2$ es un lenguaje libre de contexto.

Existen lenguajes libres de contexto L, L_1 y L_2:

- $L_1 \cap L_2$ **NO** es un lenguaje libre de contexto.
- L^c **NO** es un lenguaje libre de contexto.

Demostración

$$\begin{aligned} L_1 &= \{ a^n b^n c^m \mid n \geq 0, m \geq 0 \} \\ L_2 &= \{ a^m b^n c^n \mid n \geq 0, m \geq 0 \} \end{aligned}$$

¿son L_1 y L_2 lenguajes libres de contexto? ¿y $L_1 \cap L_2$?

Ejercicio: demuestre el caso de L^c.

Outline

Algoritmo CKY

¿cómo verificar si $w \in \mathcal{L}(\mathcal{G})$?

Dado un lenguaje libre de contexto L y una palabra w:

¿cómo verificamos si $w \in L$?

¿cómo determinar si la palabra $bbaba \in \mathcal{L}(\mathcal{G})$?

$$\begin{array}{lcl}
\mathrm{S} & \to & \mathrm{XY} \mid \mathrm{YZ} \\
\mathrm{X} & \to & \mathrm{YY} \mid \mathrm{a} \\
\mathrm{Y} & \to & \mathrm{YX} \mid \mathrm{b} \\
\mathrm{Z} & \to & \mathrm{XZ} \mid \mathrm{XX} \mid \mathrm{a}
\end{array}$$

¿cómo verificar si $w \in \mathcal{L}(\mathcal{G})$?

Dado un lenguaje libre de contexto L y una palabra w:

¿cómo verificamos si $w \in L$?

- Convertimos $\mathcal{G}$ en formal normal de Chomsky.
- Probamos todas las derivaciones de altura a lo mas $|w| + 1$.
- Si encontramos una derivación retornamos `TRUE`.

¿cuántas derivaciones debemos probar?

Algoritmo CKY

- Inventado por:

John	**C**ocke
Tadao	**K**asami
Daniel	**Y**ounger

- Algoritmo **cúbico** en $|w|$ y **lineal** en $|\mathcal{G}|$:

Tiempo: $\mathcal{O}(|w|^3 \cdot |\mathcal{G}|)$

- Un ejemplo del uso de **programación dinámica**.

Por simplicidad asumiremos que
las gramáticas esta en **Forma Normal de Chomsky** (CNF)

…CKY se puede adaptar para producciones mayores que 2

Tabla del algoritmo CKY

Para una palabra $w = a_1 a_2 \ldots a_n$ y una gramática $\mathcal{G} = (V, \Sigma, P, S)$ construimos la **tabla CKY**:

C_{15}				
C_{14}	C_{25}			
C_{13}	C_{24}	C_{35}		
C_{12}	C_{23}	C_{34}	C_{45}	
C_{11}	C_{22}	C_{33}	C_{44}	C_{55}
a_1	a_2	a_3	a_4	a_5

- Para todo $1 \leq i \leq j \leq n$ se define:

$$C_{i\,j} \;=\; \{\, X \in V \mid X \overset{\star}{\underset{\mathcal{G}}{\Rightarrow}} a_i \ldots a_j \,\}$$

Tabla del algoritmo CKY

Ejemplo

Considere la palabra *bbaba* y la gramática:

$$
\begin{array}{lcl}
S & \rightarrow & XY \mid YZ \\
X & \rightarrow & YY \mid a \\
Y & \rightarrow & YX \mid b \\
Z & \rightarrow & XZ \mid XX \mid a
\end{array}
$$

$\{S, Y\}$				
$\{S, Y\}$	$\{X, Z\}$			
$\{X, Z\}$	$\{X\}$	$\{S\}$		
$\{X\}$	$\{S, Y\}$	$\{S\}$	$\{S, Y\}$	
$\{Y\}$	$\{Y\}$	$\{X, Z\}$	$\{Y\}$	$\{X, Z\}$
b	b	a	b	a

¿cómo **verificamos** si $w \in \mathcal{L}(\mathcal{G})$ usando la tabla CKY?

¿cómo **construimos** la tabla CKY?

Algoritmo CKY: construcción de la tabla CKY

Paso 0 (inicial)

$$\begin{array}{ccccc} C_{11} & C_{22} & C_{33} & C_{44} & C_{55} \\ \hline a_1 & a_2 & a_3 & a_4 & a_5 \end{array}$$

Para cada i, construimos el conjunto $C_{ii} \subseteq V$ tal que:

$$C_{ii} \quad = \quad \left\{ X \in V \mid X \to a_i \in P \right\}$$

Algoritmo CKY: construcción de la tabla CKY

Ejemplo (Paso 0)

Considere la palabra *bbaba* y la gramática:

$$
\begin{array}{lcl}
S & \rightarrow & XY \mid YZ \\
X & \rightarrow & YY \mid a \\
Y & \rightarrow & YX \mid b \\
Z & \rightarrow & XZ \mid XX \mid a
\end{array}
$$

$\{Y\}$	$\{Y\}$	$\{X, Z\}$	$\{Y\}$	$\{X, Z\}$
b	b	a	b	a

Algoritmo CKY: construcción de la tabla CKY

Paso 1

$$
\begin{array}{|ccccc}
C_{12} & C_{23} & C_{34} & C_{45} & \\
C_{11} & C_{22} & C_{33} & C_{44} & C_{55} \\
\hline
a_1 & a_2 & a_3 & a_4 & a_5
\end{array}
$$

Para cada i, construimos el conjunto $C_{i\,i+1} \subseteq V$ tal que:

$$C_{i\,i+1} \quad = \quad \left\{ X \in V \;\middle|\; X \to YZ \in P \text{ para algún } Y \in C_{ii} \wedge Z \in C_{i+1\,i+1} \right\}$$

Algoritmo CKY: construcción de la tabla CKY

Ejemplo (Paso 1)

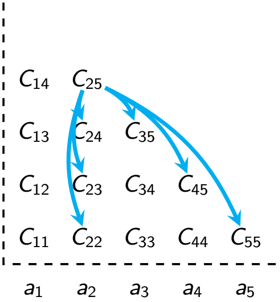
Considere la palabra *bbaba* y la gramática:

$$
\begin{array}{lcl}
S & \rightarrow & XY \mid YZ \\
X & \rightarrow & YY \mid a \\
Y & \rightarrow & YX \mid b \\
Z & \rightarrow & XZ \mid XX \mid a
\end{array}
$$

$\{X\}$	$\{S, Y\}$	$\{S\}$	$\{S, Y\}$	
$\{Y\}$	$\{Y\}$	$\{X, Z\}$	$\{Y\}$	$\{X, Z\}$
b	b	a	b	a

Algoritmo CKY: construcción de la tabla CKY

Paso k $(k > 0)$)

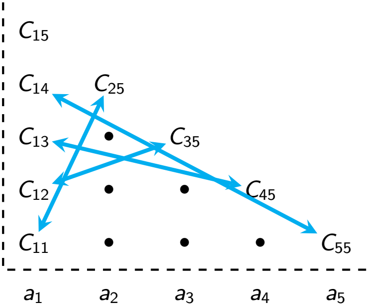


Para cada i, construimos el conjunto $C_{i\,i+k} \subseteq V$ tal que:

$$C_{i\,i+k} = \left\{ X \in V \;\middle|\; \begin{array}{ll} \exists j \in [i, i+k). & X \to YZ \in P \\ & \text{para algún } Y \in C_{ij} \wedge Z \in C_{j+1\,i+k} \end{array} \right\}$$

Algoritmo CKY: construcción de la tabla CKY

Paso k ($k > 0$)



Para cada i, construimos el conjunto $C_{i\,i+k} \subseteq V$ tal que:

$$C_{i\,i+k} \quad = \quad \left\{ \; X \in V \;\middle|\; \begin{array}{ll} \exists j \in [i, i+k). & X \to YZ \in P \\ & \text{para algún } Y \in C_{ij} \wedge Z \in C_{j+1\,i+k} \end{array} \right\}$$

Algoritmo CKY: construcción de la tabla CKY

Ejemplo (Paso k)

Considere la palabra *bbaba* y la gramática:

$$
\begin{array}{lcl}
S & \rightarrow & XY \mid YZ \\
X & \rightarrow & YY \mid a \\
Y & \rightarrow & YX \mid b \\
Z & \rightarrow & XZ \mid XX \mid a
\end{array}
$$

$\{S, Y\}$				
$\{S, Y\}$	$\{X, Z\}$			
$\{X, Z\}$	$\{X\}$	$\{S\}$		
$\{X\}$	$\{S, Y\}$	$\{S\}$	$\{S, Y\}$	
$\{Y\}$	$\{Y\}$	$\{X, Z\}$	$\{Y\}$	$\{X, Z\}$
b	b	a	b	a

Algoritmo CKY: construcción de la tabla CKY

Otro ejemplo

Considere la palabra *abba* y la gramática:

$$
\begin{array}{lcl}
S & \rightarrow & XY \mid YX \mid SS \mid \\
  &   & XZ \mid YW \\
X & \rightarrow & a \\
Y & \rightarrow & b \\
Z & \rightarrow & SY \\
W & \rightarrow & SX
\end{array}
$$

$\{S\}$			
$\{Z\}$	$\varnothing$		
$\{S\}$	$\varnothing$	$\{S\}$	
$\{X\}$	$\{Y\}$	$\{Y\}$	$\{X\}$
a	b	b	a

Algoritmo CKY

input : Una gramática $\mathcal{G} = (V, \Sigma, P, S)$ y una palabra $w = a_1 a_2 \dots a_n$
output: `TRUE` ssi $w = a_1 a_2 \dots a_n \in \mathcal{L}(\mathcal{G})$

Function `AlgoritmoCKY` *($\mathcal{G}$, w)*

    for $i \leftarrow 1$ **to** n **do**
        let $C_{ii} = \varnothing$
        for $X \to c \in P$ **do**
            if $c = a_i$ **then let** $C_{ii} = C_{ii} \cup \{X\}$
    for $k \leftarrow 1$ **to** $n - 1$ **do**
        for $i \leftarrow 1$ **to** $n - k$ **do**
            let $C_{i\,i+k} = \varnothing$
            for $j \leftarrow i$ **to** $i + k - 1$ **do**
                for $X \to YZ \in P$ **do**
                    if $Y \in C_{i\,j} \wedge Z \in C_{j+1\,i+k}$ **then let**
                      $C_{i\,i+k} = C_{i\,i+k} \cup \{X\}$
    return check $S \in C_{1n}$

Análisis algoritmo CKY

Correctitud algoritmo CKY

Para toda gramática $\mathcal{G}$ y para toda palabra $w \in \Sigma^*$ se tiene que:

$$\texttt{AlgoritmoCKY}(\mathcal{G}, w) = \texttt{TRUE} \quad \text{ssi} \quad w \in \mathcal{L}(\mathcal{G})$$

Demostración (ejercicio)

Si el input es de tamaño $|w|$ y la gramática es de tamaño $|\mathcal{G}|$, entonces:

$$\text{Tiempo Algoritmo CKY: } \mathcal{O}(|w|^3 \cdot |\mathcal{G}|)$$

¿es posible verificar si $w \in \mathcal{L}(\mathcal{G})$ en **tiempo lineal** en $|w|$?