Teoría de Lenguajes
Segunda parte
Teórica 6: Parsing de Cocke Younger Kasami

Verónica Becher

Segundo cuatrimestre 2020

Bibliografía para esta clase:
A. V. Aho, J. D. Ullman, The Theory of Parsing, Translation, and Compiling, Vol. 1 , Parsing. Prentice Hall, 1972.

`https://www-2.dc.uba.ar/staff/becher/Aho-Ullman-Parsing-V1.pdf` Capítulo 4.2

Algoritmo Cocke Younger-Kasami

Veremos ahora un algoritmo que requiere tiempo en el orden de n^3 para una entrada de longitud n. Esencialmente es un algoritmo de programación dinámica. Este algoritmo tiene interés teórico pero no tiene interés práctico porque

- Tiempo n^3
- Usa espacio n^2

y hay algoritmos más eficientes.

Forma normal de Chomsky

Definición. *Una gramática libre de contexto $G = (N, T, P, S)$ está en forma normal de Chomsky si todas sus produccionesson de la forma $A \to BC$ y $A \to a$, para $A, B, C \in N$, $a \in T$, y si $\lambda \in L(G)$ entonces $S \to \lambda$ está en P y S no aparece en la parte deracha de ninguna producción.*

Algoritmo para pasar a forma Normal de Chomsky (Aho Ullman vol 1, p.151)

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

Output GLC $G' = (N', T, P', S')$ forma normal Chomsky, $L(G) = L(G')$.

Método Poner en P'

- si $\lambda \in L(G)$ poner $S' \to \lambda$ y $S' \to S$
- todas las producciones $A \to a$ y $A \to BC$
- para cada producción $A \to X_1 X_2$ con X_1 o X_2 o ambos en T poner

$$A \to X_1' X_2'$$

para cada producción $A \to X_1 \dots X_k$ con $k > 2$, $X \in (N \cup T)$ poner

$$\begin{aligned} A &\to X_1' \langle X_2 \dots X_k \rangle \\ \langle X_2 \dots X_k \rangle &\to X_2' \langle X_3 \dots X_k \rangle \\ &\dots \\ \langle X_{k-1} \dots X_k \rangle &\to X_{k-1}' X_k \end{aligned}$$

Si $X_i \in T$, X_i' es nuevo, agregar

$$X_i' \to X_i$$

Si $X_i \in N$ entonces X_i' es igual a X_i.

Para $j = 1, 2, \dots, k-1$, $\langle X_j \dots X_k \rangle$ es nuevo no-terminal.

Observar que $|N'| \leq |N| + |T| + \ell \times |P|$ donde ℓ es el máximo número de símbolos del lado derecho de las producciones de P.

El algoritmo realiza una cantidad de operaciones lineal en $|P|$.

Algoritmo Cocke Younger-Kasami (CYK)

Sea $G = (N, T, P, S)$ libre de contexto en forma normal de Chomsky sin λ-producciones y $w = a_1 a_2 \dots a_n$ la cadena de entrada.

El algoritmo construye una tabla triangular para w

$$\mathcal{T} = (t_{i,j})_{1 \leq i \leq n, 1 \leq j \leq n-i+1}$$

Cada $t_{i,j}$ es un subconjunto de N.

$$A \in t_{i,j} \text{ si y solo si } A \overset{+}{\Rightarrow} a_i a_{i+1} \dots a_{i+j-1}$$

Por lo tanto, $w \in L(G)$ si y solo si $S \in t_{1,n}$.

Si queremos una (o todas) las derivaciones de w podemos usar la tabla $\mathcal{T}$ para construirlas.

Tabla Cocke-Younger-Kasami (CYK)

Algoritmo 1 (Table Cocke-Younger-Kasami, Algorithm 4.3 Aho Ullman vol 1)**.** Input. $G = (N, \Sigma, P, S)$ *libre de contexto en forma normal de Chomsky sin* λ*-producciones y una cadena de input* $w = a_1 a_2 \dots a_n$ *en* T^+.

Output. $\mathcal{T}$ *para* w *tal que* $t_{i,j}$ *contiene* A *si y solo si* $A \overset{+}{\Rightarrow} a_i a_{i+1} \dots a_{i+j-1}$.

Método. *Definir* $t_{i,1} = \{A : A \to a_i \in P\}$ *para* $i = 1, \dots, n$.

Si ya computamos $t_{i,j'}$, *para* $i = 1, \dots, n$ *y* $j' = 1, \dots, j-1$.

Definir

$$t_{i,j} = \{A : A \to BC \in P, \text{ para algún } k, 1 \le k < j, B \in t_{i,k} \text{ y } C \in t_{i+k,j-k}\}$$

Notar que $t_{i,k}$ *y* $t_{i+k,j-k}$ *se computan antes que* $t_{i,j}$, *ya que* $i \le k < j$, *por lo tanto,* $k < j$ *y* $j - k < j$.

Notar que si $A \in t_{i,j}$ *entonces*

$$A \Rightarrow BC \overset{+}{\Rightarrow} a_i \dots a_{i+k-1} C \overset{+}{\Rightarrow} a_i \dots a_{i+k-1} a_{i+k} \dots a_{i+j-1}$$

Repetir hasta completar $j = n - i + 1$.

Luego hay que dar una derivación de $a_1, \dots a_n$.

Correctitud tabla $\mathcal{T}$ CYK

Teorema 2 (Theorem 4.6 Aho Ullman vol 1)**.** *Sea* $G = (N, T, P, S)$ *GLC en forma normal Chomsky y sea tabla* $\mathcal{T}$ *para cadena* $a_1, \dots a_n$. *Entonces*

$$A \in t_{i,j} \text{ si y solo si } A \overset{+}{\Rightarrow} a_i \dots a_{i+j-1}$$

Demostración. Por inducción en j.
Caso base, $j = 1$. Por definición de $A \in t_{i,1}$ si y solo si $A \to a_i$
Caso inductivo, $j \ge 2$. Supongamos que vale para $j - 1$. Por defínción

$$t_{i,j} = \{A : A \to BC \in P, \text{ para algún } k, 1 \le k < j, B \in t_{i,k} \text{ y } C \in t_{i+k,j-k}\}$$

Luego,
$A \Rightarrow BC$ y, por HI, $B \overset{+}{\Rightarrow} a_i \dots a_{i+k-1}$ y $C \overset{+}{\Rightarrow} a_{i+k} \dots a_{i+k+j-k-1}$.
Por lo tanto, $A \in t_{i,j}$ si y solo si $A \overset{+}{\Rightarrow} a_i \dots a_i \dots a_{i+j-1}$ □

Complejidad tabla $\mathcal{T}$

Teorema 3 (Theorem 4.7 Aho Ullman vol 1)**.** *La construcción de la tabla* $\mathcal{T}$ *para una cadena de largo* n *es del orden de* n^3 *operaciones.*

Demostración. Debemos definir $\mathcal{T} = (t_{i,j})_{1 \le i \le n, 1 \le j \le n-i+1}$ Para fijar $t_{1,1}, t_{2,1}, \dots, t_{n,1}$ hacen falta n inspecciones en P.

Fijemos j. En el paso j debemos determinar $t_{i,j}$ para $i = 1, \dots n - j + 1$. Son en total $n - j + 1$ conjntos.

Fijemos i. Para determinar $t_{i,j}$ debemos examinar $t_{i,k}$ y $t_{i+k,j-k}$ para $k = 1, \dots, j-1$ que ya fueron computados antes. Son $2(j-1)$ conjuntos.
Usando $\sum_{j=1}^{n} j = n(n+1)/2$ y $\sum_{j=1}^{n} j^2 = n(n+1)(2n+1)/6$, tenemos

$$\sum_{j=1}^{n} (n - j + 1)2(j-1) = 2\sum_{j=1}^{n} (n+2)j - n - j^2 - 1 = n(n^2-1)/3$$

□

Derivción a izquierda CYK

Parse a izquierda Cocke-Younger-Kasami, Algorithm 4.4, Aho Ullman vol 1

Input. Gramática GLC en forma normal de Chomsky $G = (N, T, P, S)$ donde las producciones están numeradas $1, \ldots, p$, cadena de entrada $w = a_1 \ldots a_n$, y la tabla $\mathcal{T}$ para w

Output. Derivación a izquierda para w, o "error".

Método. Asumiendo que $S \in t_{1,n}$, el algoritmo debe ejecuta $gen(1, n, S)$.

Damos un procedimiento recursivo $gen(i, j, A)$ para generar una derivación corresponte a $A \overset{*}{\underset{L}{\Rightarrow}} a_i \ldots a_{i+j-i}$,

Paso 1. Si $j = 1$ y la m-ésima producción en P es $A \longrightarrow a_i$ emitir el número de producción m.

Paso 2. Si $j > 1$ y k es el mínimo tal que $1 \leq k < j$ tal que para algún $B \in t_{i,k}$ y $C \in t_{i+k,j-k}$, $A \to BC \in P$, digamoses la $m-$ésima (podría haber varias, y elegimos solo una). Emitir el número de producción m y ejecutar $gen(i, k, B)$ seguido de $gen(i + k, j - k, C)$.

Ejemplo completo CYK, (Aho Ullman vol 1, p.315)

Consideremos GLC G

$1.S \longrightarrow AA$
$2.S \longrightarrow AS$
$3.S \longrightarrow b$
$4.A \longrightarrow SA$
$5.A \longrightarrow AS$
$6.A \longrightarrow a$

La tabla $\mathcal{T}$ para *abaab*

5	A,S				
4	A,S	A,S			
3	A,S	S	A,S		
2	A,S	A	S	A,S	
1	A	S	A	A	S
	1	2	3	4	5

Como S está en $t_{1,5}$, *abaab* está en $L(G)$. Derivación a izquierda para *abaab*:
$gen(1, 5, S)$: para $k = 1$ da $1.S \to AA$ porque $A \in t_{1,1}$ $A \in t_{2,4}$. Evaluar

$gen(1, 1, A)$: da $6.A \to a$

$gen(2, 4, A)$: da para $k = 1$, $4.A \Rightarrow SA$, ya que $S \in t_{2,1}$ y $A \in t_{3,3}$. Evaluar

- $gen(2, 1, S)$: da $3.S \to b$
- $gen(3, 3, A)$: para $k = 1$ da $5.A \to AS$ ya que $A \in t_{3,1}$ y $S \in t_{4,2}$ Evaluar
  - $gen(3, 1, A)$ da $6.A \to a$
  - $gen(4, 2, S)$ para $k = 1$ da $2.S \to AS$ ya que $A \in t_{4,1}$ y $S \in t_{5,1}$. Evaluar
    - $gen(4, 1, A)$: da $6.A \to a$
    - $gen(5, 1, S)$: da $3.S \to b$

$$S \underset{L}{\Rightarrow} AA \underset{L}{\Rightarrow} aSA \underset{L}{\Rightarrow} abAS \underset{L}{\Rightarrow} abaAS \underset{L}{\Rightarrow} abaab$$

G es ambigua y *abaab* tiene más deuna derivación a izquierda.

Complejidad de CYK

Teorema 4 ((Teorema 4.8 Aho Ullman vol 1))**.** *El Algoritmo CYK con input de longitud n requiere en el orden de n^2 operaciones elementales.*

Demostración. Asuminos $G = (N, T, P, S)$ en forma normal Chomsky y tabla $\mathcal{T}$ para entrada w de longitud n. Por inducción en j, demostramos que $gen(i, j, A)$ requiere a lo sumo $c_1 j^2$ operaciones. Determnaremos la constante c_1 en la demostración.

Para $j = 1$ es trivial.

Para $j = 2$, para $k = 1, 2, \ldots j - 1$, revisar $t_{i,k}$ y $t_{i+k,j-k}$. Esto lleva $c_2 j$ operaciones, para una constante c_2. Por HI

$gen(i, k, B)$ requiere $c_1 k^2$ y $gen(i + k, j - k, B)$ requiere $c_1(j - k)^2$

$$c_1 k^2 + c_1(j - k)^2 + c_2 j = c_1(j^2 + 2k^2 - 2jk) + c_2 j$$

Como $1 \leq k < j$, y $j \geq 2$ tenemos $2k^2 - 2kj \leq 2 - 2j \leq -j$. Entonces, si tomamos c_1 igual a c_2 en HI, obtenemos que la cantidad total de operaciones en el paso j es

$$c_1(j^2 + 2k^2 - 2jk) + c_2 j \leq c_1 j^2.$$

□