Parsing CYK

a.a. 2020-2021

Corso di Fondamenti di Informatica - 1 modulo

Corso di Laurea in Informatica

Università di Roma "Tor Vergata"

Prof. Giorgio Gambosi

Riconoscimento in CFL: Cocke-Younger-Kasami

Algoritmo deterministico per il riconoscimento dei linguaggi di tipo 2: utilizza un modello di calcolo con memoria ad accesso diretto (non solo LIFO), quindi più potente di un PDA.

L'algoritmo CYK applica il paradigma della Programmazione Dinamica e, data una grammatica non contestuale in forma normale di Chomsky $\mathcal{G} = \langle V_T, V_N, P, S \rangle$ ed una stringa $x \in V_T^*$ (con $|x| = n$), determina in tempo $O(n^3)$ se x è derivabile in $\mathcal{G}$ o meno.

Riconoscimento in CFL: Cocke-Younger-Kasami

Data $x = a_1, a_2, \ldots, a_n \in V_T^+$:

- Sia $x_{i,j}$ ($1 \leq i \leq n, 1 \leq j \leq n - i + 1$) la sua sottostringa $a_i, a_{i+1}, \ldots, a_{i+j-1}$ di lunghezza j che inizia dall'i-esimo carattere di x
- Sia $A_{i,j} \subseteq V_N$ l'insieme dei simboli non terminali in $\mathcal{G}$ da cui è possibile derivare $x_{i,j}$:

$$A_{i,j} = \{A \in V_N \mid A \overset{*}{\underset{\mathcal{G}}{\Longrightarrow}} x_{i,j}\}$$

Riconoscimento in CFL: Cocke-Younger-Kasami

Osservazione:

Per ogni i ($1 \leq i \leq n$) è immediato determinare
$A_{i,1} = \{A \in V_N \mid A \longrightarrow a_i\}$ per ispezione di P.

Riconoscimento in CFL: Cocke-Younger-Kasami

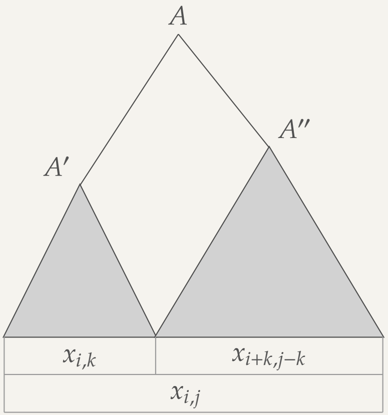
Osservazione:

Per ogni coppia i, j ($1 \leq i \leq n, 1 < j \leq n - i + 1$) un non terminale A appartiene ad $A_{i,j}$ se e solo se esiste k ($1 \leq k \leq j - 1$) tale che:

1. esiste $A' \in A_{i,k}$;
2. esiste $A'' \in A_{i+k,j-k}$;
3. esiste una produzione $A \longrightarrow A'A'' \in P$.

Infatti, in tal caso la stringa $x_{i,j}$ è derivabile da A applicando dapprima la produzione $A \longrightarrow A'A''$ e derivando quindi separatamente i suoi primi k caratteri (la sottostringa $x_{i,k}$) da A' e gli altri $j - k$ (la sottostringa $x_{i+k,j-k}$) da A''.

Riconoscimento in CFL: Cocke-Younger-Kasami



Riconoscimento in CFL: Cocke-Younger-Kasami

Per costruire l'insieme $A_{i,j}$ $(1 \leq i \leq n, 1 < j \leq n - i)$ è necessario quindi far variare il valore k tra 1 e $j - 1$ esaminando via via gli insiemi $A_{i,k}$ e $A_{i+k,j-k}$.

Chiaramente, per poter far ciò, dovremo assumere che siano disponibili tutti gli insiemi $A_{r,k}$, con $1 \leq k \leq j - 1$ e $1 \leq r \leq i + j - k$.

Infine, si noti che x è derivabile in $\mathcal{G}$ se e solo se $S \in A_{1,n}$.

Riconoscimento in CFL: Cocke-Younger-Kasami

L'algoritmo opera riempiendo un array T di dimensioni $n \times n$, in cui nella locazione $T[i, j]$ viene rappresentato l'insieme $A_{i,j}$, e verificando, alla fine di tale processo, se S compare nella locazione $T[1, n]$.

La matrice T è triangolare, in quanto non risultano definiti gli insiemi $A_{i,j}$ per $j > n - i + 1$, e l'algoritmo costruisce la matrice stessa per colonne.

Riconoscimento in CFL: Cocke-Younger-Kasami

```
for i := 1 to n do
    T[i,1] := ∅
    for each A ⟶ a ∈ P:
        if a = a_i then T[i,1] = T[i,1] ∪ {A}
for j := 2 to n − 1 do
    for i := 1 to n − j + 1 do
        T[i,j] := ∅;
        for k := 1 to j − 1 do
            for each B ∈ T[i,k] do
                for each C ∈ T[i+k, j−k] do
                    for each D ∈ V_N do
                        if D ⟶ BC ∈ P then T[i,j] := T[i,j] ∪ {D}
    if S ∈ T[1,n] then return VERO
    else return FALSO
```

Riconoscimento in CFL: Cocke-Younger-Kasami

Linguaggio L delle stringhe palindrome. Grammatica in CNF

$$
\begin{array}{lcl}
S & \longrightarrow & XZ \mid YU \mid ZZ \mid UU \\
X & \longrightarrow & ZS \\
Y & \longrightarrow & US \\
Z & \longrightarrow & 0 \\
U & \longrightarrow & 1
\end{array}
$$

Vogliamo verificare, applicando l'algoritmo CYK, se la stringa 0110 appartiene ad L.

Riconoscimento in CFL: Cocke-Younger-Kasami

Matrice T 4×4, riempita colonna per colonna, da sinistra verso destra.

Prima colonna riempita mediante l'ispezione delle produzioni che generano un terminale.

	1	2	3	4
1	Z			
2	U			-
3	U		-	-
4	Z	-	-	-

Riconoscimento in CFL: Cocke-Younger-Kasami

Il riempimento della seconda colonna, corrispondente al caso $j = 2$, risulta in:

	1	2	3	4
1	Z	$\emptyset$		
2	U	S		-
3	U	$\emptyset$	-	-
4	Z	-	-	-

Riconoscimento in CFL: Cocke-Younger-Kasami

Il riempimento della terza colonna, corrispondente al caso $j = 3$, risulta in:

	1	2	3	4
1	Z	$\emptyset$	X	
2	U	S	$\emptyset$	-
3	U	$\emptyset$	-	-
4	Z	-	-	-

Riconoscimento in CFL: Cocke-Younger-Kasami

Infine, il riempimento della quarta colonna, corrispondente al caso $j = 4$, risulta in:

	1	2	3	4
1	Z	$\emptyset$	X	S
2	U	S	$\emptyset$	-
3	U	$\emptyset$	-	-
4	Z	-	-	-

Tale situazione rappresenta il fatto che l'intera stringa, corrispondente all'elemento $T[1, 4]$, può essere derivata a partire da S.

Parsing mediante CYK

Non è difficile modificare l'algoritmo presentato in modo tale da consentire di ricostruire l'albero sintattico relativo ad una stringa riconosciuta come appartenente al linguaggio.

A tal fine, è sufficiente memorizzare, insieme ad ogni simbolo $B \in A_{i,j}$, i riferimenti ai due simboli $B_1 \in A_{i,k}$ e $B_2 \in A_{i+k,j-k}$ per i quali si ha $B \longrightarrow B_1B_2$.

Questo, per l'esempio considerato, produrrebbe la tabella estesa seguente

	1	2	3	4
1	$Z \longrightarrow 0$	$\emptyset$	$X \longrightarrow Z_{(1,1)}S_{(2,2)}$	$S \longrightarrow X_{(1,3)}Z_{(4,1)}$
2	$U \longrightarrow 1$	$S \longrightarrow U_{(2,1)}U_{(3,1)}$	$\emptyset$	-
3	$U \longrightarrow 1$	$\emptyset$	-	-
4	$Z \longrightarrow 0$	-	-	-