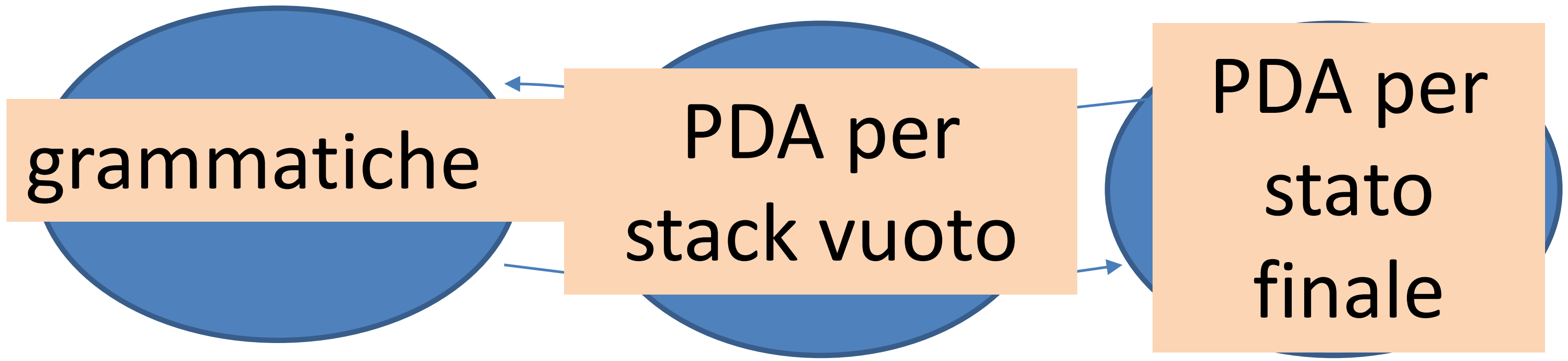
Lezione 5

CFG e PDA definiscono gli stessi linguaggi



la direzione più intuitiva va dalle grammatiche ai PDA per stack vuoto

si costruisce un PDA che simula le derivazioni leftmost ogni forma sentenziale lm non terminale è:

$xA\alpha$

--x è stringa di terminali

--A è la variabile più a sinistra

-- α è stringa di variabili e di terminali

$A\alpha$ è la <u>coda</u> della forma sentenziale

Intuizione:

se $S \Rightarrow^*_{lm} xA\alpha \Rightarrow^*_{lm} w$, con $w=xw'$

allora il PDA

$(p,w,Z0) \vdash^* (p,w',A\alpha) \vdash (p,\varepsilon,\varepsilon)$

e se $A \rightarrow \beta$, allora

$\delta(p, \varepsilon, A) = \{(p, \beta)\}$

e se β contiene terminali ? Facciamo match con l'input il PDA “indovina” una derivazione da S e usa i terminali che produce per consumare l’input

se l’input finisce e lo stack si vuota (coda della forma sentenziale)
allora ha “indovinato” la derivazione giusta

sono tutte derivazioni leftmost.

costruzione: sia $G=(V,T,P,S)$, costruiamo $P=(\{q\}, T, V \cup T, \delta, q, S)$, dove δ è come segue:

1) per ogni variabile A in V, $\delta(q,\varepsilon,A)=\{(q,\beta) \mid A \rightarrow \beta$ è una produzione in $P\}$
2) per ogni a in T, $\delta(q,a,a)=\{(q,\varepsilon)\}$ per consumare l'input

esempio:

I -> a | b | Ia | Ib | I0 | I1

E -> I | E*E | E+E | (E)

non determinismo usato pesantemente

I terminali del PDA sono {a,b,0,1,(,),+,*}, questi assieme a E e I sono i simboli dello stack.

a) $\delta(q, \varepsilon, I)=\{(q,a),(q,b),(q,Ia),(q,Ib),(q,I0),(q,I1)\}$
b) $\delta(q,\varepsilon,E)=\{(q,I),(q,E*E),(q,E+E),(q,(E))\}$
c) $\delta(q,a,a)=\{(q,\varepsilon)\}$, $\delta(q,b,b)=\{(q,\varepsilon)\}$, $\delta(q,0,0)=\{(q,\varepsilon)\}$, $\delta(q,1,1)=\{(q,\varepsilon)\}$, $\delta(q,(,()=\{(q,\varepsilon)\}$, $\delta(q,*,*)=\{(q,\varepsilon)\}$ ..ecc.

Teorema 6.13

Se P è il PDA costruito da G allora N(P)=L(G)

Dimostrazione: w è in N(P) sse w è in L(G)

(<=) supponiamo che esista $S=\gamma 1 \Rightarrow^*lm\ \gamma 2 \Rightarrow^*lm\ \gamma 3 \Rightarrow^*lm \dots \Rightarrow^*lm\ \gamma n=w$

Per induzione su i in [1..n] dimostriamo che (q,w,S)|-*(q,yi,αi), tale che γi=xi αi e xi yi=w, quindi xi è quello che è stato consumato dell'input e yi quello che resta da consumare.

Base: i=1, γ1=S, quindi x1=ε e y1=w. Dato che (q,w,S)|-*(q,w,S) in 0 mosse, la base è ok

induzione: assumiamo che sia vera per i >=1, cioè che (q,w,S)|-*(q,yi,αi), con γi=xi αi e w=xi yi e dimostriamo che vale per i+1.

αi = Aη, e in γi=xi Aη =>lm xi $\beta\eta$ usando la produzione A -> β

per costruzione c'è

δ(q,ε,A)={(q, β)} e quindi

(q,yi,A η)|- (q, yi, β η) se in $\beta\eta$ ci sono terminali li elimina tutti consumando yi fino ad ottenere nello stack la coda di γi+1 così si arriva a αn che è la coda di γn=w e quindi αn=ε per cui abbiamo che (q,w,S)|-(q, ε, ε) quindi P accetta w

(=>) se P esegue una serie di mosse che ha l'effetto di eliminare un A dalla cima della pila senza mai toglierlo in precedenza, allora A genera la stringa di input consumata nel frattempo:
se (q,x,A) |-* (q, ε, ε) allora A =>* x

per induzione sul numero di mosse di P

Base: una mossa, $(q, x, A) \vdash^* (q, \varepsilon, \varepsilon)$

Dalla costruzione in G deve esserci $A \to \varepsilon$
quindi $x = \varepsilon$ e $A \Rightarrow \varepsilon$

Induzione: supponiamo che P faccia n mosse con n>1
La prima mossa deve essere del tipo (1) che corrisponde alla produzione
$A \to Y_1 \dots Y_k$

quindi $(q,x,A) \vdash (q,Y_1 \dots Y_k)$ dopo di che n-1 mosse eliminano $Y_1 \dots Y_k$ e x possiamo scomporre $x=x_1 \ldots x_k$ dove x_1 è la parte di input consumata finché si elimina Y_1, x_2 quella consumata finche si elimina Y_2 e così via.

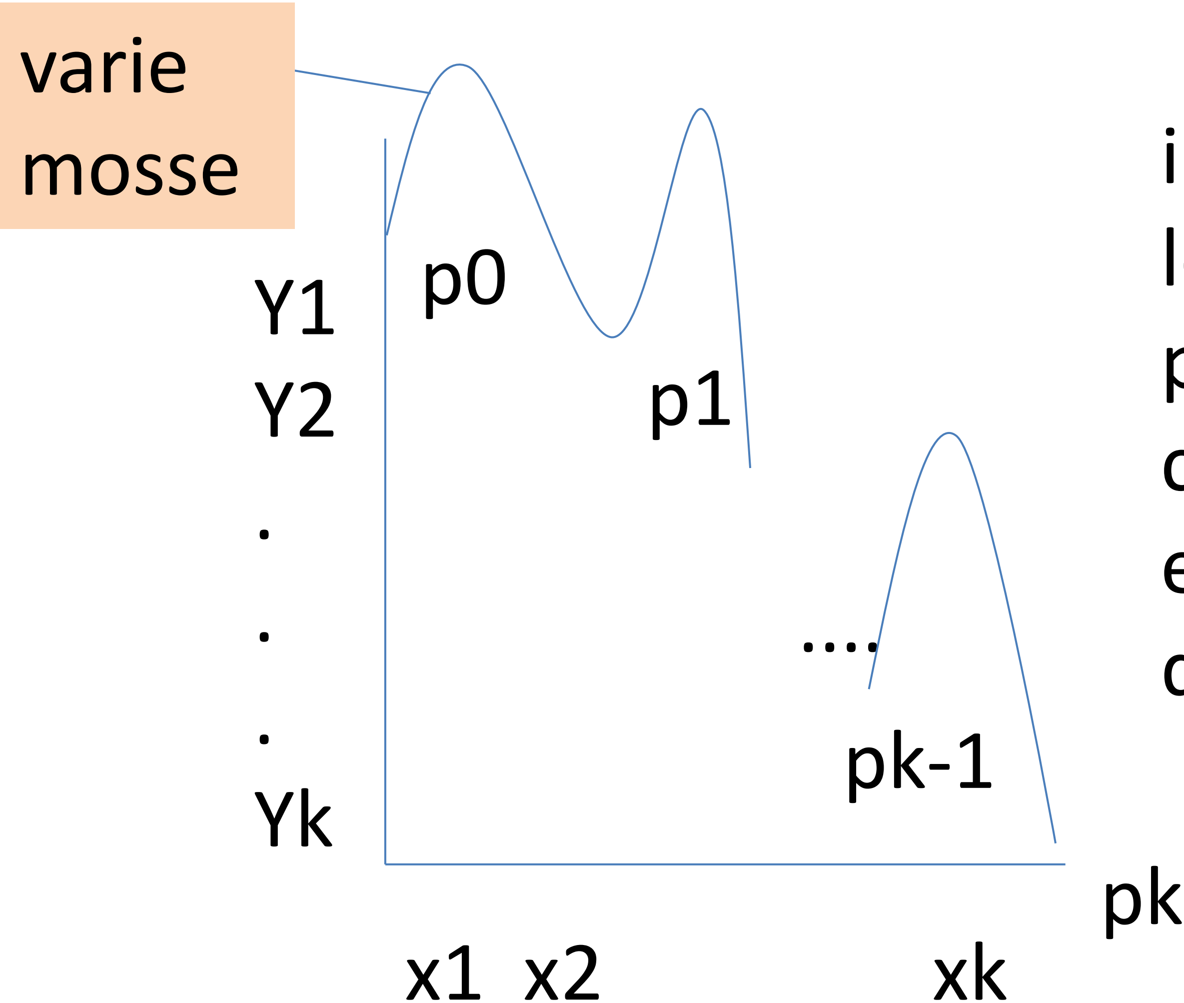
Se Y_i è terminale allora la sua eliminazione avviene attraverso una mossa (2) che consuma Y_i dall'input

quindi $(q, x_i \ldots x_k, Y_i) \mid\!-^* (q, x_{i+1} \ldots x_k, \varepsilon)$ per $i=1,2,..k$ e ognuna di queste computazioni è più corta di n e quindi, per ipotesi induttiva,

$Y_i \Rightarrow^* x_i$ e quindi $A \Rightarrow Y_1 \ldots Y_k \Rightarrow^* x_1 \ldots x_k = x$

se $S=A$ e $w=x$, da $(q,w,S) \mid\!-^* (q, \varepsilon, \varepsilon)$ segue che $S \Rightarrow^* w$

ora consideriamo da PDA → CFG



importano
le mosse in cui si fa
pop di Y1...Yk
con stati p0..pk-1
e consumo
dell’input x1...xk

la grammatica ha variabili che rappresentano
--l’eliminazione di un simbolo X dallo stack
--il passaggio dallo stato p allo stato q dopo l’eliminazione di X

queste variabili saranno [p X q]

Dato $P=(Q,\Sigma,\Gamma,\delta, q0, Z0)$ costruiamo $G=(V, \Sigma,R,S)$, dove V contiene:

1. il simbolo iniziale S
2. [p X q] per ogni X in Γ e p e q in Q

R contiene

a) S ->[q0 Z0 p] per ogni p in Q,

[q0 Z0 p] di G deve generare tutte le stringhe w che sono accettate da P con stack vuoto, cioè le stringhe consumate fino a che Z0 viene eliminato (svuotando lo stack)

(b) supponiamo che δ(q, a, X) contenga (r, Y1...Yk), dove a in Σ U {ε} e k>=0

allora, per ogni r1...rk in Qm R contiene [q X rk] -> a [r Y1 r1][r1 Y2 r2]...[rk-1 Yk rk]

questa produzione di G corrisponde alla computazione di P che arriva ad avere lo stack corrente meno X e deve generare la stringa terminale che P consuma in questa computazione funziona !!

[q X p]=>* w sse (q, w, X) |-* (p, ε, ε)

(<=) per induzione sulla computazione,
Base: 1 mossa, w deve essere un terminale o ε, per cui [q X p] -> w è in R e [q X p] => w

Induzione: supponiamo che (q, w, X) |-* (p, ε, ε) in n passi con n>1, la prima mossa deve avere la forma,
(q, w, X) |-* (r0, x, Y1...Yk) |-* (p, ε, ε) dove w=a x, per a in Σ U {ε}

dalla fig precedente: siano r1...rk gli stati quando Y1...Yk sono eliminati e w1...wk= x i pezzi di x consumati,

quindi (ri-1, wi, Yi) |-* (ri, ε, ε) e per ipotesi induttiva [ri-1 Yi ri] =>* wi

ed esiste in G:

[q X rk] -> a [r0 Y1 r1][r1 Y2 r2]....[rk-1 Yk rk]

quindi [q X rk]=>*a w1 w2 ...wk = w

(=>) per induzione sul numero di passi della derivazione.

Base: 1 passo, [q X p] -> w allora esiste δ(q, a, X) che contiene (p, ε) con a=w e e quindi (q,w,X) |- (p, ε, ε)

Induzione: assumiamo che [q X p] =>* w in n passi, dopo il primo passo,
[q X rk] => a [r0 Y1 r1][r1 Y2 r2]...[rk-1 Yk rk] =>* w
con rk=p e quindi per ogni i=1,2,...k
[ri-1 Yi ri]=>*wi con a w1 w2...wk=w

per ipotesi induttica, per ogni i=1,2,..k
(ri-1, wi Yi) |-* (ri, ε, ε)

e quindi per Teorema 6.5 vale anche
(ri-1, wiwi+1..wk YiYi+1...Yk) |-* (ri, wi+1..wk, Yi+1...Yk)

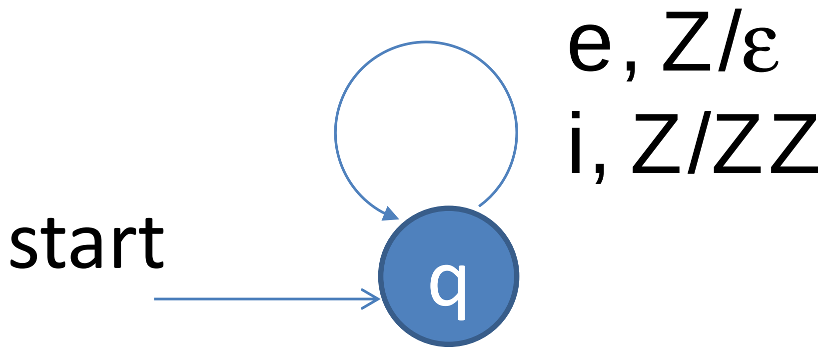
ma la produzione [q X rk] => a [r0 Y1 r1][r1 Y2 r2]...[rk-1 Yk rk] deriva dal fatto che (r0 Y1 Y2 ...Yk) è in δ(q,a,X) per cui
(q, aw1...wk,X) |- (r0,w1...wk, Y1...Yk) |-*

(r1, w2..wk,Y2..Yk) |-*(r2, w3...wk,Y3..Yk) |-* (rk, ε, ε) con rk=p

per finire basta osservare, per il punto (a) della costruzione, che S=>* w sse per qualche p, [q Z0 p]=>* w

da questo, la prova precedente dimostra che
S=>*w sse (q,w,Z0) |-* (p,ε,ε)

esempio: $P=(\{q\}, \{i,e\}, \{Z\},\delta,q,Z)$



ha solo 2 variabili: S e [q Z q]

S -> [q Z q]

[q Z q] ->i [q Z q][q Z q]

[q Z q] -> e

S -> [q Z q]

[q Z q] ->i [q Z q][q Z q]

[q Z q] -> e

S -> A

A -> i A A | e

S -> i S S | e