

Lezione 6

automi a pila deterministici

come riconoscere se un automa a pila è deterministico:

per ogni q , X e a in $\Sigma \cup \{\varepsilon\}$ deve avere al massimo una mossa possibile

o

$\delta(q, a, X) = \{(p, \alpha)\}$ per a in Σ

o

$\delta(q, \varepsilon, X) = \{(p, \alpha)\}$

I DPDA (per stato finale) accettano tutti i linguaggi regolari

Teorema 6.17

Dato un linguaggio regolare L , esiste un automa a stati finiti che lo riconosce e allora esiste anche un automa a pila che l'accetta con stato finale (basta non usare lo stack)

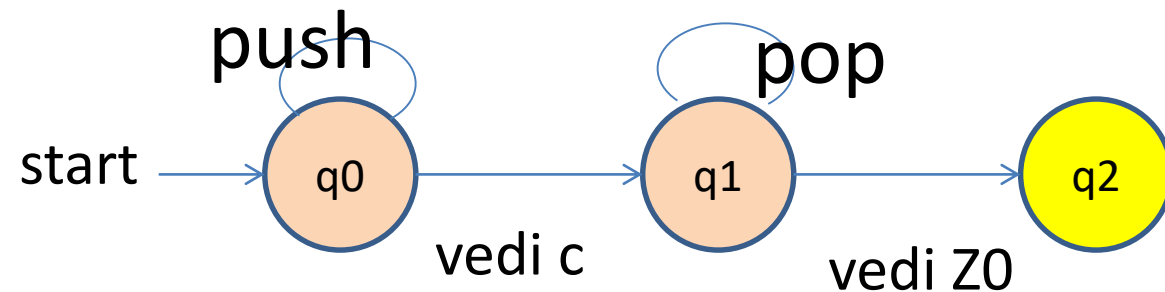
Ma possono fare di più?

Sì

esiste un DPDA che accetta con stati finali capace di accettare $wc w^r$

1) $wc w^r$ non è un linguaggio regolare: usiamo il pumping lemma con $|w| > n$, quindi $w = xyz$ e xy sta prima di c per cui $xy^k z$ non avrà lo c al centro e quindi non sarà in $wc w^r$

2) il DPDA per $wc w^r$ è:



ma i DPDA non accettano tutti i linguaggi liberi da contesto

non esiste un DPDA che accetta ww^r

intuizione:

supponiamo che ci sia un tale DPDA e che
esamini $0^n 1 10^n$

dopo i primi n 0, lo stack deve contenere
l'informazione su n , poi deve verificare che dopo
11, seguano altri n 0, e per farlo deve consumare
l'informazione sullo stack e quindi, se la stringa
fosse $0^n 1 10^n 0^n 1 10^n$ non riuscirebbe a
riconoscerla

I linguaggi riconosciuti per stato finale da DPDA sono strettamente di più dei linguaggi regolari, ma strettamente di meno di quelli liberi da contesto

e i DPDA che accettano per pila vuota?

Hanno capacità limitata: possono riconoscere solo linguaggi che hanno la proprietà del prefisso, cioè non ci sono x e w nel linguaggio tali che una sia prefisso dell'altra, cioè, per esempio $w = x w$

0^n è regolare e non ha la proprietà del prefisso e quindi non è riconosciuto per pila vuota da alcun DPDA

Teorema 6.19

Un linguaggio L è $N(P)$ per un DPDA P se e solo se L ha la proprietà del prefisso ed $L=L(P')$ per un DPDA P' .

Dimostrazione: esercizio 6.4.3

DPDA e ambiguità

osserva che ww^r ha una grammatica non ambigua:

$$S \rightarrow 0S0 \mid 1S1 \mid \varepsilon$$

quindi i DPDA non riconoscono nemmeno tutti i linguaggi liberi da contesto non inerentemente ambigui

Teorema 6.20

Ogni linguaggio L tale $L=N(P)$ per un DPDA P è generato da una CFG non ambigua

Dimostrazione: la grammatica della costruzione del Teorema 6.14 è non ambigua se si applica ad un DPDA.

ogni mossa $\delta(q,a,A)=(r,Y_1...Y_k)$ è la sola applicabile, ma da vita a tante produzioni $[q A p] \rightarrow_a [r Y_1 r_1][r_1 Y_2 r_2]...[r_{k-1} Y_k p]$ una per ogni sequenza $r_1...r_{k-1}$, OK, ma il determinismo garantisce che ci sia una sola sequenza $r_1...r_{k-1}$ che funzioni

ma che fine fanno le produzioni “sbagliate”?

non derivano w

Teorema 6.21 se $L=L(P)$ per un DPDA P , allora L ha una CFG non ambigua.

Dimostrazione sappiamo costruire grammatiche solo a partire da DPDA che accettano per stack vuoto, quindi da P costruiamo P' che accetta L per stack vuoto, ma dobbiamo fare in modo che L abbia la proprietà del prefisso

$L'=L\$$ cioè mettiamo una sentinella $\$$ alla fine di ogni stringa di L

L' ha la proprietà del prefisso ed è accettato da DPDA per stato finale \Rightarrow esiste P' t.c. $N(P')=L'$
 \Rightarrow CFG G' deterministica

ma noi vogliamo L non L' !!

da G' otteniamo G trattando $\$$ come un nonterminale e aggiungendo $\$ \rightarrow \varepsilon$

G' è non ambigua \Rightarrow anche G lo è: $\$$ ha solo la produzione $\$ \rightarrow \varepsilon$, come potrebbe introdurre ambiguità?

esercizi 6.4.1, 6.4.2 (a) e (b), e 6.4.3