Linguaggi e Traduttori

Prof. Marco Gavanelli

21 luglio 2016

Esercizio 1 (4 punti)

Si consideri il linguaggio L sull'alfabeto $\{a, b\}$ tale che L contiene tutte e sole le stringhe che hanno esattamente due a (ed un qualunque numero di b, in qualunque ordine).

Si definiscano una grammatica ed un corrispondente automa riconoscitore per il linguaggio L. Lo studente cerchi di trovare una grammatica di livello più basso possibile nella classificazione secondo Chomsky, intendendo il livello 3 come minimo e il livello 0 come massimo.

Esercizio 2 (10 punti)

Si consideri la grammatica $G = \langle \{a, b, d, f\}, \{A, B, C\}, P, A \rangle$, dove:

$$P = \begin{array}{l} A \to AB \mid a \\ B \to bC \mid d \\ C \to dCa \mid f \end{array}$$

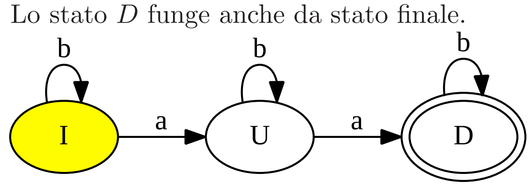
1. Si classifichi la grammatica secondo Chomsky.
2. La grammatica è LR(0)? Se sì, si scriva l'automa a stati finiti ausiliario del PDA riconoscitore. Se no, si motivi il perché.
3. La grammatica è LL(1)? Se sì, si scriva la parsing table del PDA riconoscitore. Se no, si motivi il perché.
4. Qualora la grammatica non sia LL(1), se ne scriva una equivalente G' che sia LL(1)
5. Si scriva la parsing table associata alla grammatica G'. Qualora non si sia ottenuta una grammatica LL(1), si scriva comunque la parsing table, evidenziando i conflitti

Qualora nei punti precedenti si sia riusciti a ottenere uno o più riconoscitori deterministici, si mostri come gli automi riconoscono le stringhe `abdfad` e `add`, mostrando l'evoluzione dello stack.

Soluzione 1

Il linguaggio può essere generato da una grammatica di tipo 3 (regolare).

Il seguente automa riconoscitore utilizza lo stato per memorizzare il numero di a che sono state lette. Avremo quindi uno stato Uno ed uno stato Due (oltre allo stato Iniziale). Lo stato D funge anche da stato finale.



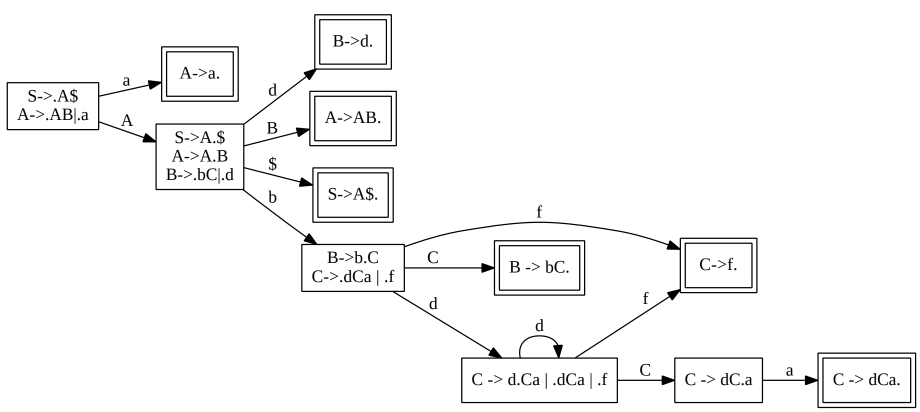
Dall'automa si può ottenere una grammatica, ad esempio regolare a destra:

$$\begin{aligned} I &\rightarrow bI \mid aU \\ U &\rightarrow bU \mid aD \mid a \\ D &\rightarrow bD \mid b \end{aligned}$$

Soluzione 2

1. La grammatica è di tipo 2 (context-free).

2. La grammatica è LR(0), infatti l'automa non presenta conflitti:



3. La grammatica non è LL(1), come si vede dal fatto che c'è una ricorsione sinistra per il simbolo A.

4. La grammatica originaria:

$$\begin{aligned} A &\rightarrow AB \mid a \\ B &\rightarrow bC \mid d \\ C &\rightarrow dCa \mid f \end{aligned}$$

Eliminiamo la ricorsione sinistra, introducendo il nuovo nonterminale D

$$\begin{array}{l} A \to aD \mid a \\ B \to bC \mid d \\ C \to dCa \mid f \\ D \to BD \mid B \end{array}$$

La grammatica non è LL(1) in quanto le due produzioni di A iniziano con lo stesso simbolo terminale a. Raccogliamo a fattor comune, introducendo il non terminale E

$$\begin{array}{l} A \to aE \\ B \to bC \mid d \\ C \to dCa \mid f \\ D \to BD \mid B \\ E \to D \mid \epsilon \end{array}$$

Anche le due produzioni per D hanno lo stesso prefisso. Raccogliamo a fattor comune, introducendo il non terminale F

$$\begin{array}{l} A \to aE \\ B \to bC \mid d \\ C \to dCa \mid f \\ D \to BF \\ E \to D \mid \epsilon \\ F \to D \mid \epsilon \end{array}$$

I due nonterminali E ed F hanno esattamente la stessa definizione, quindi possiamo eliminare uno dei due

$$\begin{array}{l} A \to aE \\ B \to bC \mid d \\ C \to dCa \mid f \\ D \to BE \\ E \to D \mid \epsilon \end{array}$$

(volendo, anche il non terminale D può essere eliminato, sostituendo la sua definizione nelle regole per E, ottenendo $E \to BE \mid \epsilon$. Questo semplifica ulteriormente i passaggi successivi).

5. Per verificare se la nuova grammatica è LL(1) e scrivere la parsing table, scriviamo gli insiemi FIRST e FOLLOW dei simboli non terminali

	$FIRST$	$FOLLOW$
A	$\{a\}$	$\{\$\}$
B	$\{b, d\}$	$\{b, d, \$\}$
C	$\{d, f\}$	$\{b, d, a, \$\}$
D	$\{b, d\}$	$\{\$\}$
E	$\{\epsilon, b, d\}$	$\{\$\}$

La parsing table è la seguente:

	a	b	d	f	$\$$
A	$A \to aE$				
B		$B \to bC$	$B \to d$		
C			$C \to dCa$	$C \to f$	
D		$D \to BE$	$D \to BE$		
E		$E \to D$	$E \to D$		$E \to \epsilon$

La parsing table non presenta conflitti, per cui la grammatica è LL(1).

Riconoscimento stringhe con automa LL(1):

Input	Stack
abdfad$	A$
abdfad$	aE$
bdfad$	E$
bdfad$	D$
bdfad$	BE$
bdfad$	bCE$
dfad$	CE$
dfad$	dCaE$
fad$	CaE$
fad$	faE$
ad$	aE$
d$	E$
d$	D$
d$	BE$
d$	dE$
$	E$
$	$

Input	Stack
add$	A$
add$	aE$
dd$	E$
dd$	D$
dd$	BE$
dd$	dE$
d$	E$
d$	D$
d$	BE$
d$	dE$
$	E$
$	$

Riconoscimento stringhe con automa LR(0):

Input	Stack
abdfad$	
bdfad$	a
bdfad$	A
dfad$	Ab
fad$	Abd
ad$	Abdf
ad$	AbdC
d$	AbdCa
d$	AbC
d$	AB
d$	A
$	Ad
$	AB
$	A
	A$
	S

Input	Stack
add$	
dd$	a
dd$	A
d$	Ad
d$	AB
d$	A
$	Ad
$	AB
$	A
	A$
	S