Linguaggi e Traduttori – Tempo: 2 ore

Prof. Marco Gavanelli

14 giugno 2018

Esercizio 1 (3 punti)

Si consideri il linguaggio $L = \{a^n(bc)^n | n > 0\} \cup \{a^k b^k c | k > 0\}$.

Si scriva una grammatica non ambigua che genera il linguaggio L; si fornisca la grammatica di livello più basso possibile nella classificazione secondo Chomsky (intendendo il livello 3 come minimo e il livello 0 come massimo).

Si classifichi la grammatica secondo Chomsky.

Se è possibile, si mostri l'albero di derivazione della stringa *abc*.

Esercizio 2 (6 punti)

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

$$P = \begin{array}{l} S \to aA \mid Ba \\ A \to bSd \mid Aa \\ B \to db \end{array}$$

1. Si classifichi la grammatica secondo Chomsky.
2. La grammatica è LL(1)? Se sì, si scriva la parsing table del PDA riconoscitore. Se no, si motivi il perché.
3. La grammatica è SLR(1)? È LR(0)? Se sì, si rappresentino gli automi riconoscitori. Se no, si motivi il perché.
4. Qualora nei punti precedenti si sia riusciti ad ottenere un automa, si mostri il riconoscimento delle stringhe *abdbad* e *abdbada* mostrando l'evoluzione dello stack.

Esercizio 3 (4 punti)

Si consideri la seguente grammatica:

$$\begin{array}{l} S \to Ab \mid Ba \mid a \\ A \to Ab \mid Sb \\ B \to Sa \end{array}$$

Si disegni l'automa riconoscitore del linguaggio generato. L'automa è deterministico? Si scriva un'espressione regolare che rappresenta il linguaggio generato dallo scopo S.

Soluzione 1

Si noti come la stringa abc sia in comune ad entrambi gli insiemi con cui il linguaggio è definito; la grammatica più immediata
$G = \langle \{a, b, c\}, \{S, A, B\}, P, S \rangle$

$$P = \begin{array}{l} S \to A \mid Bc \\ A \to aAbc \mid abc \\ B \to aBb \mid ab \end{array}$$

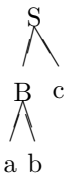
risulta quindi ambigua, in quanto ci sono due derivazioni canoniche (ad esempio, a sinistra) per la stringa abc:

- $S \to A \to abc$
- $S \to Bc \to abc$

Per avere una grammatica non ambigua bisognerà togliere la stringa abc dal linguaggio generato da A o da Bc. Scegliamo ad esempio di eliminarla da $L(A)$:

$$P = \begin{array}{l} S \to A \mid Bc \\ A \to aAbc \mid aabcbc \\ B \to aBb \mid ab \end{array}$$

La grammatica è di tipo 2 (context free).
L'albero di derivazione:

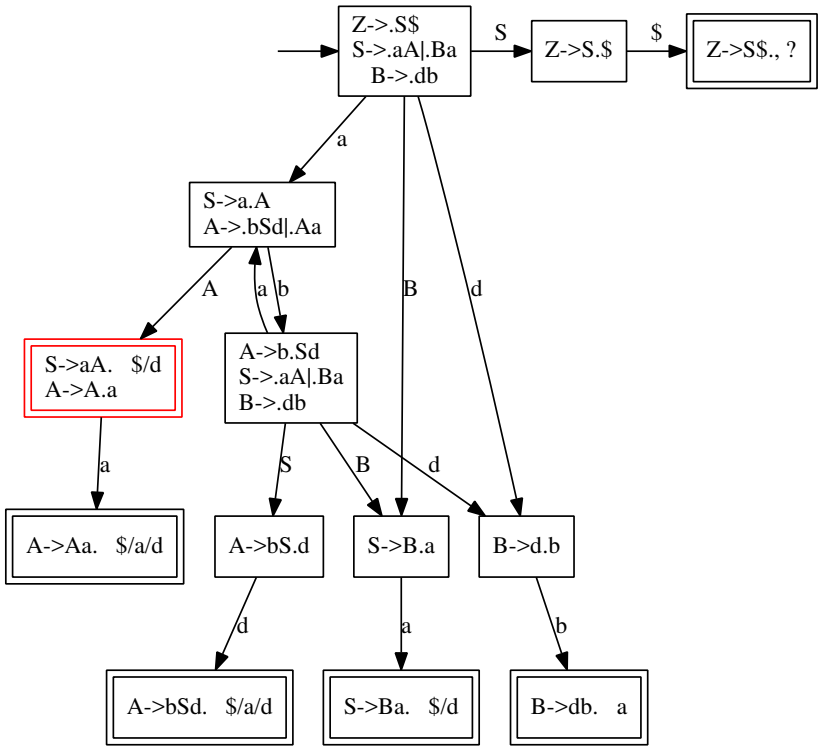


Soluzione 2

1. La grammatica è di tipo 2 (context-free).
2. La grammatica non è LL(1) in quanto contiene la ricorsione a sinistra nella produzione $A \to Aa$.
3. Per scrivere l'automa SLR(1), serve conoscere l'insieme FOLLOW dei non terminali; calcoliamo quindi gli insiemi FIRST e FOLLOW:

	$FIRST$	$FOLLOW$
S	$\{a, d\}$	$\{\$, d\}$
A	$\{b\}$	$\{\$, a, d\}$
B	$\{d\}$	$\{a\}$

L'automa SLR(1) risulta:



L'automa non presenta conflitti, quindi la grammatica è SLR(1).

Osservando il nodo indicato in rosso, si può vedere che la grammatica non è LR(0). Infatti in quel nodo l'automa SLR(1) riesce a distinguere se fare shift o reduce in base al prossimo carattere di input:

- se il prossimo carattere è \$ oppure d, effettua *reduce* $S \to aA$,
- mentre se il prossimo carattere è a, effettua *shift* a.

L'automa LR(0) non ha la possibilità di osservare il prossimo carattere di input, quindi in questo nodo si avrebbe un conflitto shift-reduce. La grammatica, quindi, non è LR(0).

Riconoscimento.

Stack	Input
	abdbad$
a	bdbad$
ab	dbad$
abd	bad$
abdb	ad$
abB	ad$
abBa	d$
abS	d$
abSd	$
aA	$
S	$
S$	
Z	

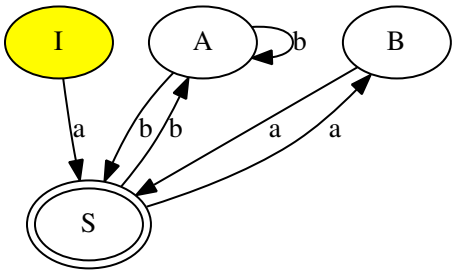
Stringa riconosciuta.

Stack	Input
	abdbada$
a	bdbada$
ab	dbada$
abd	bada$
abdb	ada$
abB	ada$
abBa	da$
abS	da$
abSd	a$
aA	a$
aAa	$
aA	$
S	$
S$	
Z	

Stringa riconosciuta.

Soluzione 3

L'automa riconoscitore:



Questo automa non è deterministico in quanto dallo stato A con input b si può passare agli stati A e S.

Per calcolare il linguaggio generato tramite espressioni regolari, riscriviamo la grammatica come sistema di equazioni:

$$\begin{cases} S = Ab + Ba + a \\ A = Ab + Sb \\ B = Sa \end{cases}$$

Sostituiamo la B nelle altre equazioni

$$\begin{cases} S = Ab + Saa + a \\ A = Ab + Sb \end{cases}$$

Eliminiamo la ricorsione diretta per A

$$\begin{cases} S = Ab + Saa + a \\ A = Sbb^* \end{cases}$$

Sostituiamo la A nell'equazione per S

$$S = Sbb^*b + Saa + a$$

Raccogliamo la S a destra del simbolo di uguaglianza:

$$S = S(bb^*b + aa) + a$$

A questo punto si può eliminare la ricorsione diretta per S:

$$S = a(bb^*b + aa)^*$$

Ovviamente, altre espressioni regolari possono descrivere lo stesso linguaggio; ad esempio

$$S = a(bbb^* + aa)^*$$