

1) Data la seguente grammatica libera da contesto  $G : S \xrightarrow{} aS | aSbS | \epsilon$ , dimostrare che il linguaggio  $L(G)$  contiene solo stringhe tali che ogni loro prefisso abbia un numero di a almeno pari al numero dei b.

$$1) G: S \xrightarrow{} aS | aSbS | \epsilon$$

$$L = \{ w \mid w \in \{a,b\}^* \text{ per ogni prefisso di } w, \#a \geq \#b \}$$

$$L(G) \subseteq L$$

Base:  $\# \text{ di derivazione.} = 1$

$$S \xrightarrow{} \epsilon, \quad \epsilon \in L \text{ perché } \#a = \#b = 0.$$

INDUZ.  $\# \text{ di derivaz. } n+1$

Abbiamo 2 casi:

$$a) S \xrightarrow{} aS \xrightarrow{m-1} a\omega' = w$$

per HP. induuttiva  $\omega' \in L$ , allora  $a\omega' \in L$  perché

$\#a \geq \#b$  per  $\omega'$ , allora  $\#a+1 \geq \#b$ , quindi

$$w \in L.$$

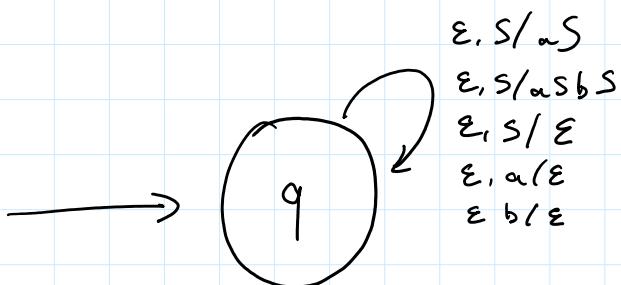
$$b) S \xrightarrow{} aSbS \xrightarrow{P} \underbrace{a\omega'}_{\substack{\text{cc} \\ \text{cc}}} bS \xrightarrow{q} a\omega'b\omega'' = w$$

$$P+q=m$$

Per HP. induuttiva  $\omega'$  e  $\omega'' \in L$  per cui aggiungere un a e un b mantiene la stringa nel linguaggio.

2)

2) Costruire un automa a pila P che accetta lo stesso linguaggio  $L(G)$  generato dalla grammatica G del punto precedente.



P.S. visto che abbiamo la grammatica dall'es. 1,

Convertiamo la CFB in PDA, con i passaggi spiegati nel libro, a pag. 225

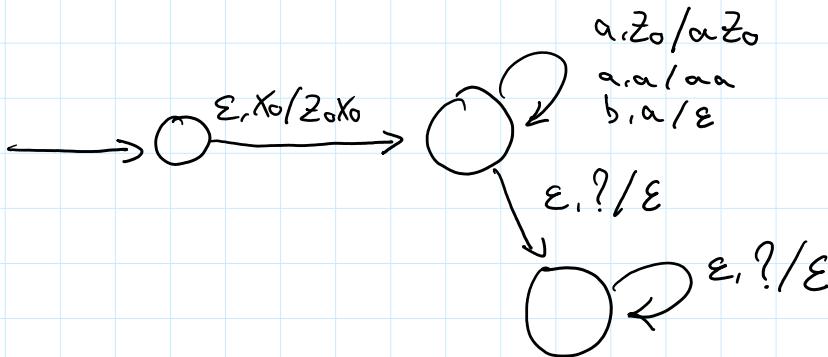
3)

In generale gli automi a pila possono accettare per pila vuota o per stati finali. I linguaggi riconosciuti sono gli stessi. Per gli automi a pila deterministici questo non è più vero. Spiegate le ragioni di questa differenza. Cercate di specificare quali linguaggi vengono accettati nelle due modalità di accettazione. La differenza tra le 2 classi di linguaggi vi pare importante?

Sol. reperibile nei file del Tutorato del 22/05

4)

Dato l'automa a pila  $P = (\{q\}, \{a,b\}, \{a,Z\}, \delta, q, Z, \{q\})$  dove  $\delta$  è come segue:  $\delta(q, a, Z) = \{(q, aZ)\}$ ,  $\delta(q, a, a) = \{(q, aa)\}$ ,  $\delta(q, b, a) = \{(q, \epsilon)\}$ . Descrivere il linguaggio riconosciuto da  $P$ . Trasformare  $P$  in un PDA  $P'$  che accetta per pila vuota lo stesso linguaggio accettato da  $P$  per stato finale.



P.S Con "?" si indica tutti i simboli possibili.

Il PDA riconosce lo stesso linguaggio già dell'es. 1.

5)

5) Data la seguente grammatica libera da contesto  $G : B \rightarrow BB \mid (B) \mid \epsilon$ , rispondere alle seguenti due domande:

a) dimostrare, per induzione sulla lunghezza della derivazione, che  $L(G)$  consiste di stringhe in  $\{(, )\}^*$  in cui le parentesi siano bilanciate, cioè tali che ogni parentesi ( ha una corrispondente ) che la segue e se la coppia di parentesi venisse eliminata, si otterrebbe di nuovo una stringa bilanciata.

b) Mostrare che la grammatica  $G : B \rightarrow (B) \mid \epsilon$ , non genera tutte le stringhe in  $\{(, )\}^*$  bilanciate.

$$G: B \rightarrow BB \mid (B) \mid \epsilon$$

a) Base.  $n=1$

$$B \rightarrow \epsilon, \text{EL. OK.}$$

INDUZ.  $n+1$  passo.

due sotto casi:

$$B \rightarrow (B) \xrightarrow{n-1} (w') = w$$

per HP. induttiva,  $w' \in L$ , per cui  $(w') = w \in L$ .

$$B \rightarrow BB \xrightarrow{P} w' B \xrightarrow{q} w' w'' = w$$

$$\text{con } P+q=n,$$

Per HP. induttiva,  $w' \in L$ ,  $w'' \in L$ , per cui anche  $w \in L$

b)  $G: B \rightarrow (B) \mid \epsilon$ .

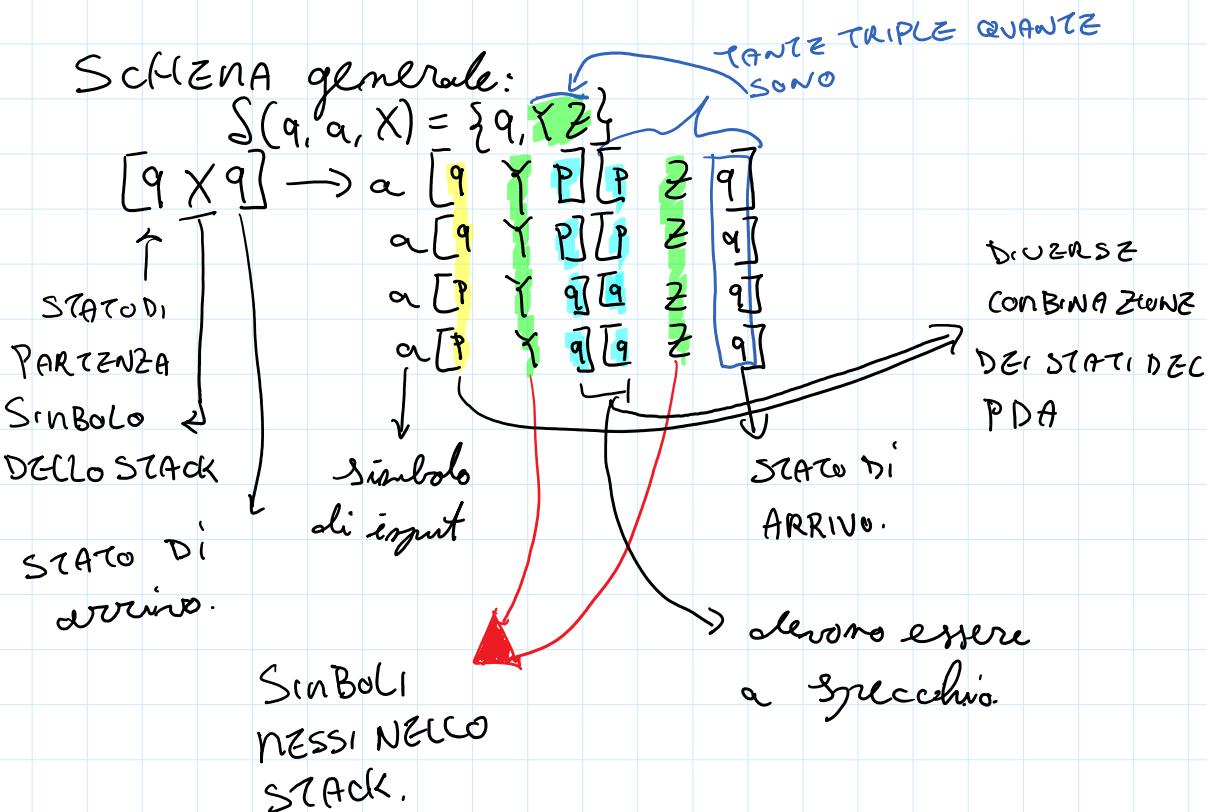
La grammatica non può produrre la parola: ( ) ( )

6)

La domanda riguarda la dimostrazione che per ogni PDA  $P$  che accetta per stack vuoto, esiste una grammatica  $G$  che genera lo stesso linguaggio che  $P$  riconosce. Ora, immaginate che  $P$  abbia solo 2 stati ( $q$  e  $p$ ) e che abbia la seguente transizione:  $\delta(q, a, X) = \{(q, Y Z)\}$ . Mostrare le produzioni che  $G$  possiede in corrispondenza di questa transizione.

$$S(q, a, X) = \{q, Y Z\}$$

$$\begin{aligned} [q \times q] &\rightarrow a [q \quad Y \quad P] [P \quad Z \quad q] \\ &\quad a [q \quad Y \quad P] [P \quad Z \quad q] \\ &\quad a [P \quad Y \quad q] [q \quad Z \quad q] \\ &\quad a [P \quad Y \quad q] [q \quad Z \quad q] \end{aligned}$$



7)

7) Data la seguente CFG:  $S \rightarrow ASB | \epsilon$ ,  $A \rightarrow aAS | a$ ,  $B \rightarrow SbS | A | bb$ , descrivere come si eliminano da essa le  $\epsilon$ -produzioni, ottenendo una grammatica che genera  $L(S) - \{\epsilon\}$ .

$$\begin{aligned} S &\rightarrow ASB | \epsilon \\ A &\rightarrow aAS | a \\ B &\rightarrow SbS | A | bb \end{aligned}$$

$$Z = \{S\}$$

$$\begin{aligned} S &\rightarrow ASB | AB \\ A &\rightarrow aA | aAS | a \\ B &\rightarrow Sb | bS | SbS | b | A | bb \end{aligned}$$

P.S.

Creo l'insieme  $Z$  dove ci metto tutte le produzioni che hanno  $\epsilon$ .

Per ogni produz. che ha solo simboli in  $Z$  aggiungo il simbolo  $a$  in  $Z$ .

Poi sistemo le produzioni assente dei simboli in  $Z$ .

8) Th. di Rice dice che tutte le proprietà non banali sui linguaggi RE sono indecidibili.  
Sia PCF: proprietà context free

Definire PCF che è indecidibile per Rice.

Spiegare quali proprietà dei linguaggi RE sono dette banali.

Sol:

$$PCF = \{ L \mid L \text{ rispetta PCF} \}$$

Una proprietà si dice Banale se è soddisfatta da tutti i linguaggi oppure da nessuna.

9) Descrivere CFN

Descrivere PL per CFL.

Spiegare come si dimostra PC.

Sol:

Una grammatica si dice in CFN se rispetta le seguenti condizioni:

- Non vuoto
- Ogni produzione è di questa forma  $A \rightarrow BC$  con A, B, C variabili.
- $A \rightarrow a$  con A variabile e a terminale
- non ha simboli inutili

PL:  
Sia  $L$  un CFL, allora esiste una costante  $n$  tale che, se  $z$  è una stringa in  $L$ , con  $|z| \geq n$  allora possiamo scrivere  $z = uvwxy$  tali che

1)  $|vwx| \leq n$  (mediana con lung. limitata)

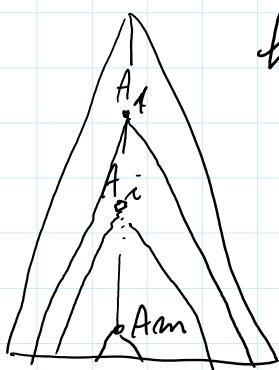
2)  $VX \neq \emptyset$

3) Per ogni  $i \geq 0$ ,  $uv^iw^jx^y$ , ovvero le due stringhe  $vx$  possono essere replicate mantenendo  $uv^iw^jx^y$  in  $L$ .

Per CNT trasforma albero sintattico delle grammatiche non CN in una grammatica equivalente che abbia un albero sintattico binario.

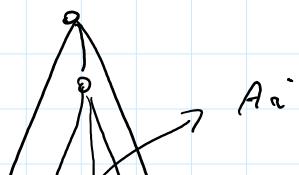
Se l'albero ha l'altezza  $n$ , allora la parola generata  $w$ , ha  $|w| \leq 2^{n-1}$

Supponiamo di avere una CFG in CN, con  $m$  variabili in CFG, allora lunghezza della parola prodotta con l'albero sintattico



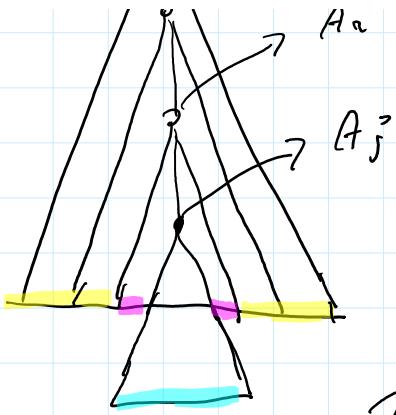
ha la lunghezza massima  $2^{m-1}$ ,

per cui per generare le parole con lunghezza  $\geq 2^{m-1}$ , alcune variabili/proiezioni vengono usate almeno 2 volte, come nell'albero:



con  $A_i = A_j$ ,

albero con PC, la suddivisione



allora con PL, la suddivisione  
uVWXy ha:

u e y sono rispettivamente lunghe

prima e la parte dopo di  $A_i$ .

$W$  è il prodotto di  $A_j$ ,

mentre le due parti intermedi sono

$V$  e  $X$  che possono essere replicate,  
perché basta usare più volte la suddivisione  
 $A_i$ .

10)

Usare il pumping lemma per dimostrare che  $L = \{a^nbnc^i \mid i \leq n\}$  non è CF. Vedere esercizi 7.2.1 del testo

PL è ovviamente  $\Leftrightarrow$  la parola da provare  
CF è almeno  $\geq h$ .

Sia  $h$  la lunghezza della PL e la parola

$a^h b^h c^h \in L$

$\exists$  una suddivisione della parola in  $uVWXy$

t.c.  $|VWX| \leq h$

$|VX| \neq \emptyset$ .

Per cui possiamo trovare più casi.

a)  $a^h b^h c^h$

$VX = b^p$  comp, allora con  $k=0$  si ha  $a^h b^{h-p} c^h$

$h \neq h-p$ , quindi  $\notin L$

b)  $a^h b^h c^h$

VX comprende sia degli a che b,  $\#a = p$ ,  $\#b = q$  con  $p \neq q > 0$ , quindi con  $k=0$ , abbiamo:

$a^{h-p} b^{h-q} c^h$

$h-p < h$  perché  $p > 0$ .  $\#a < \#c$

oppure

$h-q < h$  perché  $q > 0$ .  $\#b < \#c$ .

c)  $a^h b^h c^h$

VX comprende  $P \xrightarrow{a^k} a^h b^h c^h$  con  $k=0$ ,

altrago:

$a^{h-p} b^h c^h \notin L$  perché  $h-p < h$   $p > 0$

d)  $a^h b^h c^h$

VX comprende alcuni b ( $\#q > 0$ ) e alcune c ( $\#P$ )

quindi con  $k=0$ , altrago:

$a^h b^{h-q} c^{h-p} \notin L$

$h \neq h-q$  per  $q > 0$ .

e)  $a^h b^h c^h$

VX contiene  $c^p$  con  $p > 0$ , e con  $k=2$  ha

$a^h b^h c^{h+p} \notin L$

$h < h+p$  per  $p > 0$

11)

Descrivere un PDA che accetta per pila vuota ed è capace di riconoscere il linguaggio  $L = \{(ab)^n (ca)^m \mid n \geq 1\}$ . Il vostro è un automa deterministico o nondeterministico? Spiegare la risposta.

$$L = \{(ab)^n (ca)^m \mid n \geq 1\}$$

Nisto che costruire il PDA per questo non è facile, mentre scrivere la sua grammatica b è, scriviamola.

per poi convertirla in PDA. Con gli stessi procedimenti dell'es. 2.

$$G: S \rightarrow abSca \mid abc a$$

CONVERSIONE:

$$\delta(q, a, a) = \{q, \epsilon\}$$

$$\delta(q, b, b) = \{q, \epsilon\}$$

$$\delta(q, c, c) = \{q, \epsilon\}$$

$$\delta(q, \epsilon, S) = \{(q, abSca), (q, abc a)\}$$

Il PDA ottenuto non è deterministico perché l'ultima transizione ha due risultati.

12)

Dare la definizione del linguaggio LU e spiegare in dettaglio come si dimostra che LU è un linguaggio RE e non ricorsivo

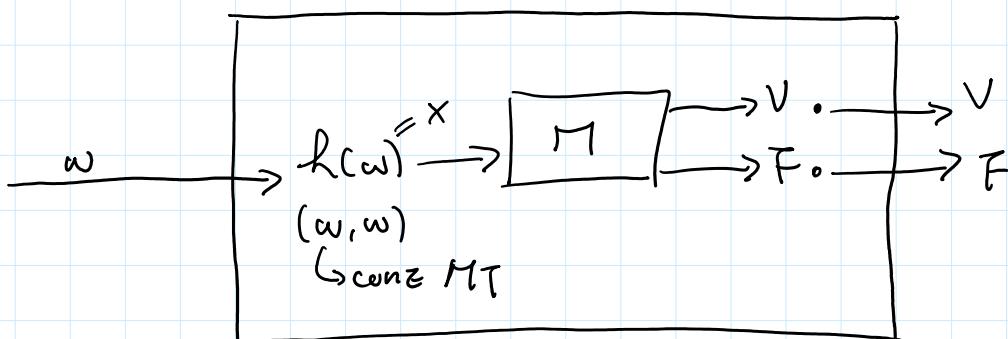
$$L_M = \left\{ (M, w) \mid w \in \{0, 1\}^* \text{ e } M \text{ T M, tale che} \right. \\ \left. M \text{ accetta } w \right\}$$

Din:

$L_M$  è RE, simulo  $w$  su  $M$

Supponiamo che non sia RE per assurdo

Sia  $L_M$  ricorsivo,  $\overline{L_M}$  è ricorsivo  $\rightarrow$  è anche RE  
riduciamo  $L_D$  a  $\overline{L_M}$



assurdo

$L_D$  è il linguaggio di Gödel della diagonalizzazione, ovvero

$w = M$ ) ma  $M$  non accetta  $w$  per def.

- 13) Spiegare cos'è una TM multi-track. Come esempio di uso di queste TM, spiegare la simulazione delle TM multi-nastro con le TM multi-track e spiegare quanti passi deve fare la multi-track per simulare  $n$  passi della multi-nastro e perché.

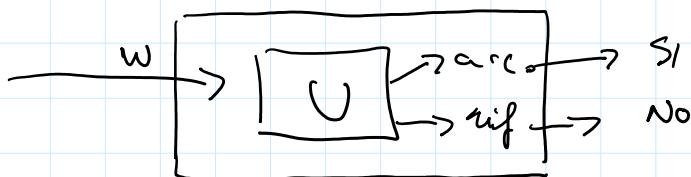
Pag. 318

- 14) Definire i linguaggi  $L_e$  e  $L_{ne}$  e spiegare dove si situano nel diagramma dei linguaggi ricorsivi, RE e non RE. Spiegare la risposta.

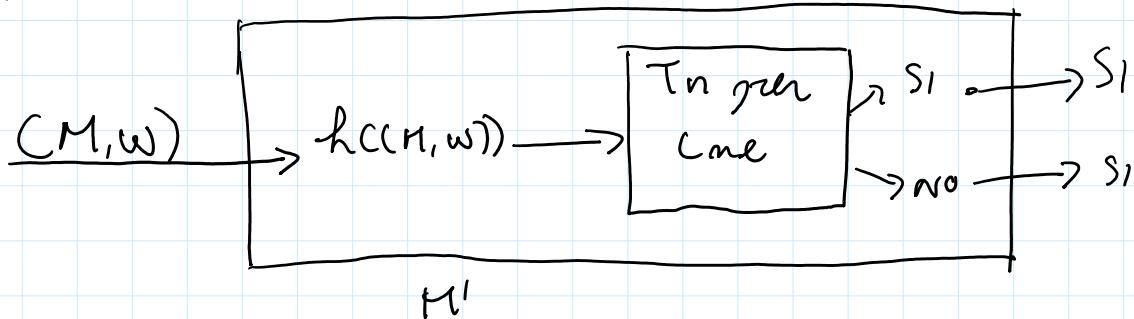
$$L_e = \{M \mid L(M) = \emptyset\}$$

$$L_{ne} = \{M \mid L(M) \neq \emptyset\}$$

$L_m$  è RE perché esiste un M.T. f.c



$L_{ne}$  non è ricorsivo:



$L_{ne}$  è RE ma non ricorsivo,  $h((n, w))$  è  $T_n$ ,  
ignora l'input e parla  $w$  su  $M$

$M'$  parla  $w$  su  $M$  ignorando input,  $M'$  è rifiutata  
quando  $M$  non accetta  $w$ .

Non è rifiutata quando  $M$  accetta  $w$ , per cui  $M'$   
è ricorsivo  $\rightarrow L(M)$  è ricorsivo.