

Descrivere, anche tramite un disegno, la relazione che esiste tra i linguaggi regolari, i linguaggi CF, i linguaggi CF non inerentemente ambigui, i linguaggi riconosciuti dai PDA che accettano per stato finale, quelli riconosciuti dai PDA che riconoscono per pila vuota e, per finire, quelli riconosciuti dai DPDA che accettano per stato finale e quelli riconosciuti dai DPDA che accettano per pila vuota. La risposta va motivata in modo conciso.

Tutte queste caratterizzazioni sono composte da linguaggi che accettano un insieme comune di operazioni chiuse, descrivendo principalmente unione, concatenazione, chiusura di Kleene, ecc.

Le varie operazioni sono definite induttivamente, tali che:

- nei linguaggi regolari danno luogo ad un automa, tale che esista almeno un ciclo (PL)
- nei linguaggi CF, analogamente, esiste una sottoderivazione ricorsiva (PL per i CF)
- nei linguaggi per PDA, essi avranno necessariamente uno stato iniziale, svuotato alla fine della pila ed un insieme di stati finali, tali che le transizioni regolari siano composte da push/pop degli stati tolti alla fine (pila vuota).

Tutti quindi presentano stati ripetuti, scegliibili deterministicamente/indeterministicamente, tali da poter avere ripetizione di stringhe e scelta possibile a seconda delle proprietà del linguaggio.

2. Descrivere un PDA, che accetta per pila vuota, che riconosca il seguente linguaggio $L = \{a^n b^m \mid 0 \leq n \leq m \leq 2n\}$. E' possibile costruire il PDA passando prima per una CFG che genera L. In questo caso è richiesta una dimostrazione o almeno una spiegazione convincente del fatto che la CFG generi veramente L.

Il linguaggio è costituito da un numero diverso di "a" e "b", avendo in particolare un numero doppio come massimale di "a" rispetto a quello delle "b".

L'ordine di queste stringhe deve essere mantenuto, avendo idealmente un numero di "a" uguale a quelle delle "b" facendo pop di un singolo per volta, oppure eseguendo il pop di due "b" e ottenendo $m=2n$ oppure alternando le cose, avendo quindi un numero doppio di "a" rispetto alle "b" nelle normali condizioni.

Partiamo dalla descrizione di una grammatica che descrive questo linguaggio:

$S \rightarrow aA \mid \epsilon$

$A \rightarrow aAbB \mid \epsilon$

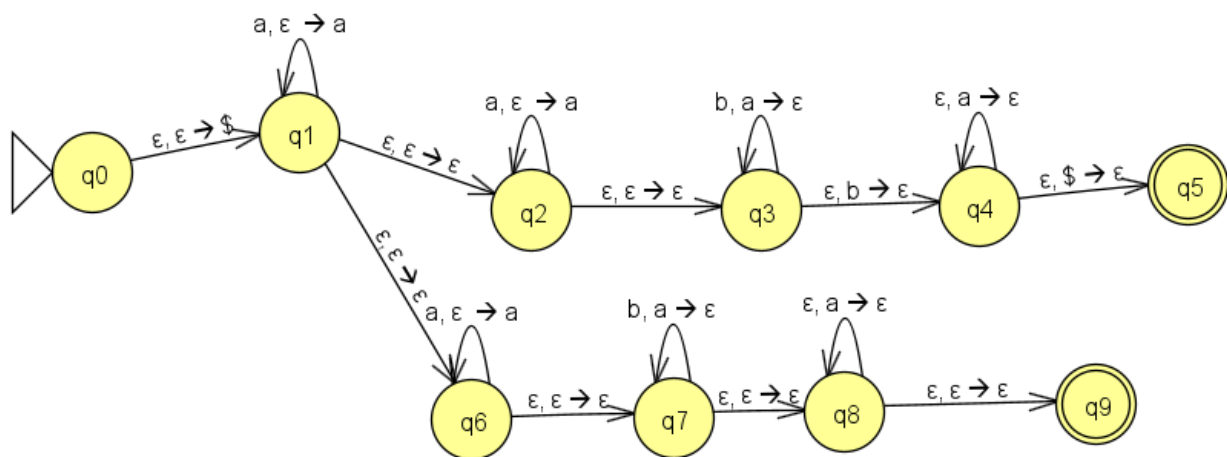
$B \rightarrow aB \mid \epsilon$

In questo modo posso avere già almeno un numero doppio di "a" rispetto alle "b" e la CFG genera effettivamente questo linguaggio, avendo ad esempio come derivazione:

$S \rightarrow aA \rightarrow aAbB \rightarrow aAbaB \rightarrow aAbBaB \rightarrow aab$

Segue il possibile PDA di questa grammatica:

(seguendo l'idea numero di "a"=numero di "b" oppure per l'idea di prima numero doppio di "a" rispetto al numero di "b")



Si chiede di descrivere la Forma Normale di Chomsky (FNC) e di descrivere l'enunciato del pumping Lemma, spiegando, infine, come il fatto che ogni CFG possa venire messa in FNC sia fondamentale per dimostrare il pumping Lemma.

La forma normale di Chomsky si pone l'idea di avere regole formate da derivazioni con al massimo 2 transizioni, di cui ve ne è almeno una terminale. Essendo che ogni linguaggio context-free è dimostrabile essere generato da una formale normale di Chomsky, ci si pone l'obiettivo di trasformarlo:

- aggiungendo una nuova variabile iniziale
- eliminazione delle ϵ -regole
- eliminazione regole unitarie $A \rightarrow B$
- trasformazione delle regole rimaste per avere solo 2 transizioni finali

Similmente, il pumping lemma si propone, nel caso dei linguaggi CF, di avere una parola di lunghezza $|w| \geq k$ con una suddivisione $w=uvxyz$ tali che possiamo pompare uv^ixy^iz per un $i \geq 0$

Si nota quindi che il PL sia dimostrato a causa del fatto che, essendo una grammatica ricorsiva, stanti le condizioni di prima abbiamo bisogno di almeno 2 simboli ripetuti ed 1 terminale, avremo almeno una singola ripetizione delle parole (seguendo l'idea del $B=2^V + 1$ del pumping lemma, nel caso in cui almeno uno dei due simboli sia terminale, pur avendo regole vuote, da cui il +1)