CS-303

MID-SEMESTER

ASSIGNMENT

- TARUSI MITTAL

- 1901CS65

-

Que 1:

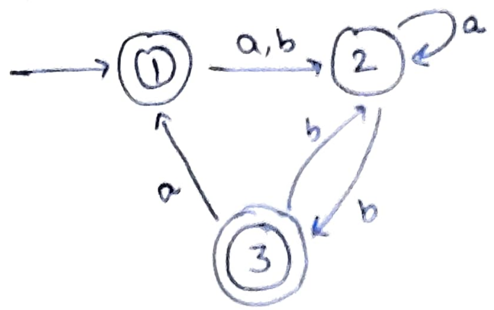
(a) $\Sigma = \{0,1\}$, $L = \{w \mid w$ contains an even number of 0's or w contains exactly two 1's$\}$

Ans: $1^*(01^*01^*) + (0^*10^*10^*)$

(b) $L = \{\varepsilon, 0\}$

Ans: $0 + \varepsilon$

(c)



Language generated by DFA

Ans: Here we have two accepting states: {1, 3}, therefore we need to reach either of them starting from 1

Now, we will make RE r_{13} to reach state 3 starting from state 1.

$\therefore r_{13} = \underbrace{(a+b)}_{\text{to move to state 2 from state 1}} \underbrace{(a^*(bb)^*)^*}_{\text{to move to state 3 from state 2}} b$

We can reach state 1 again ~~fro~~ by making a cycle via RE $(r_{13} \cdot a)$

So, if we want to finally end up at state 1, we can do that by $r_{13} \cdot a + \lambda$
$\hookrightarrow$ by not moving anywhere.

$\therefore$

Ans: -

$$\boxed{\begin{gathered} r = (r_{13}a)^* (\lambda + r_{13}), \text{ where} \\ r_{13} = (a+b)(a^*(bb)^*)^* b \end{gathered}}$$

(d) $\Sigma = \{a,b\}$, $L = \{w \mid w$ has an odd number of a's and ends with ab$\}$

Ans: $b^* (ab^* ab^*)^* ab$

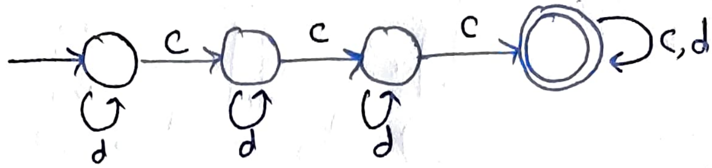
(e) $L = \{w \mid w$ is any string except 11 and 111$\}$, $\Sigma = \{0,1\}$

Ans: $(\varepsilon + 1) + (0 + 10 + 110 + 1110 + 1111)(0+1)^*$

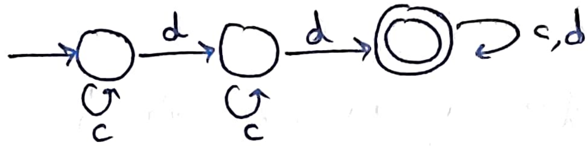
Que 2:- Give DFA for the following language in $\{c, d\}^*$

(a) $\{w \mid w$ has at least three c's and at least two d's$\}$

Ans: DFA for at least 3 c's

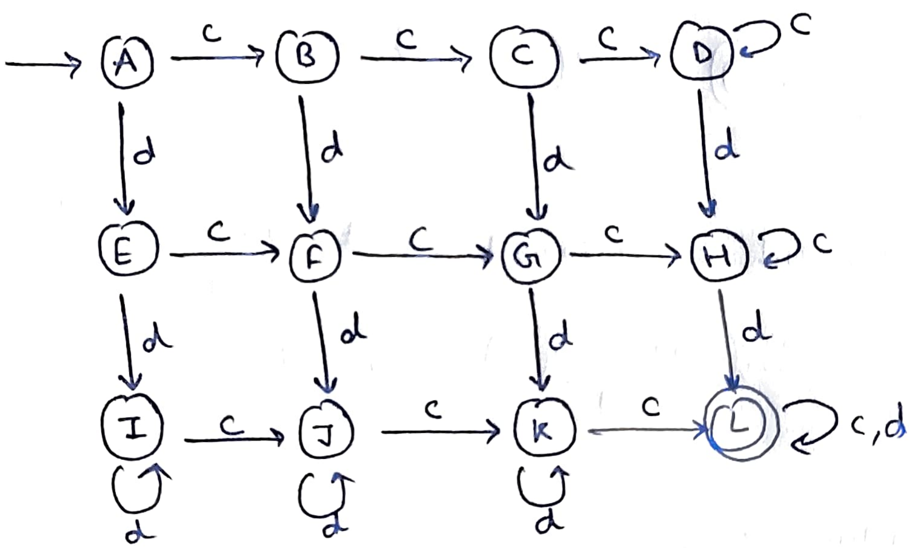


DFA for at least 2d's



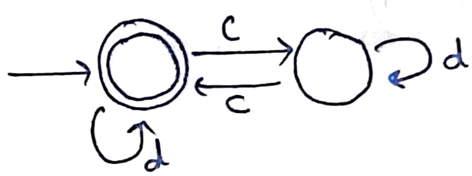
On taking interaction

DFA ⇒

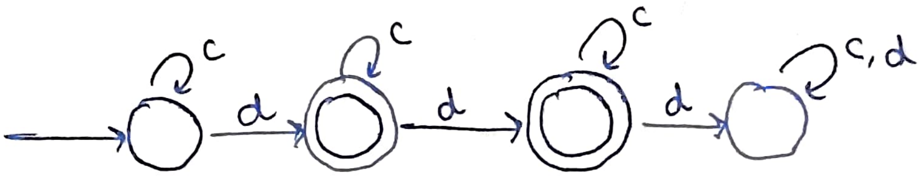


(b) $\{w \mid w$ has even no. of c's and one or two d's$\}$

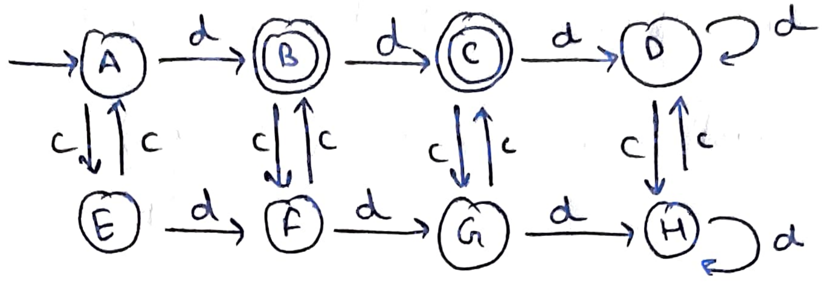
Ans: DFA for even c's



DFA for one or two d

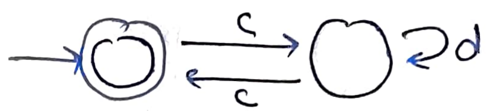


Intersection ⇒

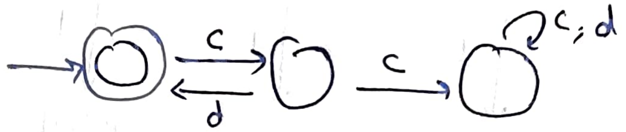


(c) {w|w has an even no's of c's and each c is followed by at least one d.}

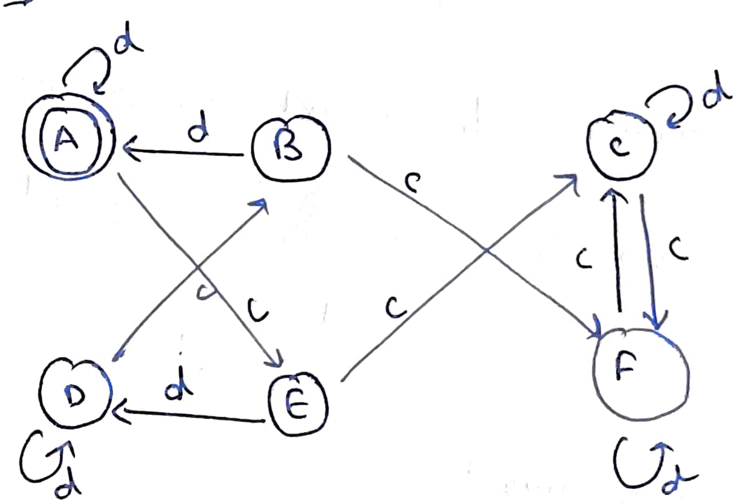
DFA for even c's



DFA for each c followed by at least one d

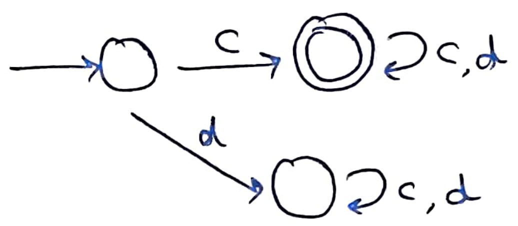


Interaction ⇒

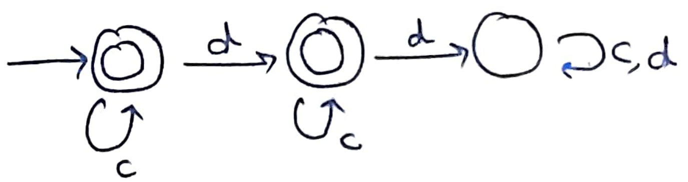


(d) {w | w start with an c and has at most one d}

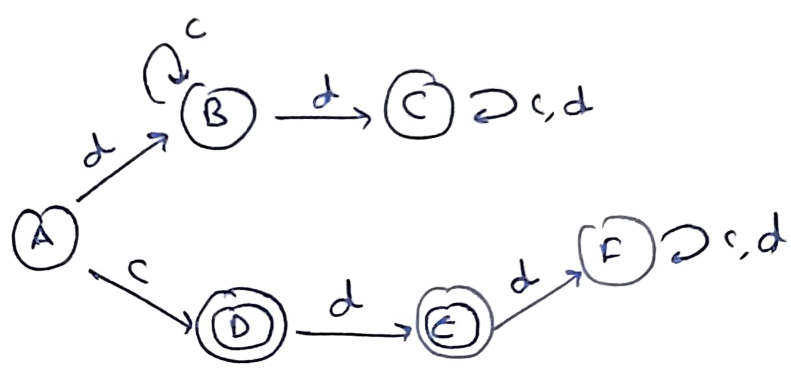
DFA for start with c.



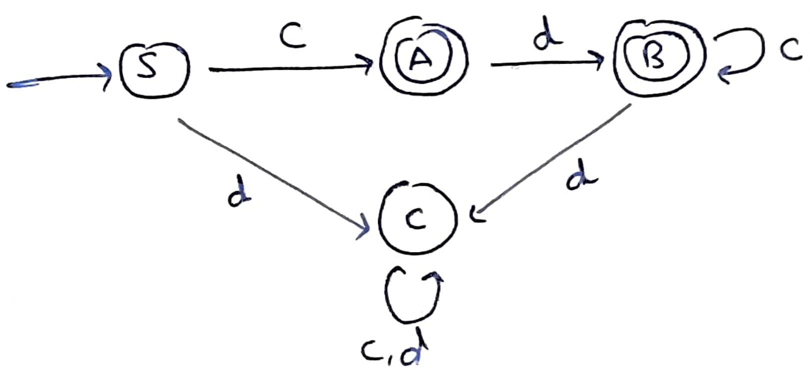
DFA for at most one d.



Intersection:



Reduced:



Que 3: For any language A over Σ, consider the language of strings obtained by deleting a single character from any string in A.

$$\text{Delete}(A) = \{xz \mid x, z \in \Sigma^* \text{ and } xyz \in A \text{ for some } y \in \Sigma\}$$

Show that if A is regular, then Delete(A) is regular.

Ans: Let A be regular language and D a DFA that recognizes it.
Now, we will construct an NFA N than recognizes Delete(A) (let it be $P(N)$)

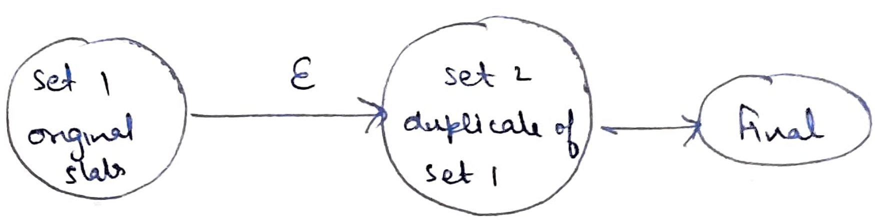
Now the basic idea of our NFA N is to have a second set of states, because they are matching to DFA, D.
Here also it has an option to choose a λ transition, effectively guessing as to what the deleted characters might while there will be no λ transition in second set of states indicating that one character has already been deleted.

Let $D = (Q, \Sigma, \delta, q_0, F)$

$N = (Q \cup Q', \Sigma, \delta', q_0, F')$

here, $Q' = \{q' \mid \text{where } \forall q \in Q \ (q)' = q'\}$

$F' = \{q' \mid \text{where } \forall q \in F \ (q)' = q'\}$



Schematic View of NFA

$\rightarrow \delta'$ is given by $\delta'(q,c) = \{\delta(q,c)\}$

$\delta'(q',c) = (\delta(q,c))'$

$\delta(q,\varepsilon) = \{(\delta(q,a))' \mid a \in \Sigma\}$

$\delta'(q',\varepsilon) = \phi$

Now all we have to do is that prove $P(N) =$ delete (A).

(1) $P(N) \subseteq$ Delete (A)

Let $s \in P(N)$, then by defination we know that, $s = s_1 s_2 s_3 \dots s_n$ where $s_i \in \Sigma_\varepsilon$ and we have $t_0 t_1 \dots t_n$, where $t_0 = q_0 =$ start state of DFA, D and $t_n \in F'$ and each $t_{i+1} \in \delta'(t_i, s_{i+1})$.

Now, $t_n \in F' \subseteq Q'$, we will choose the smallest k such that $t_k \in Q'$, Thus $t_i \in Q'$ $\forall\ i \geq k$.

Since there is no transition from set Q' to set, Q.
The following inferences can be deduced.

a) $s_i \neq \epsilon$ for $i > k$, since no $\delta'(t_j, s_{j+1})$ can be empty of $0 \leq j < n$

b) $s_i \neq \epsilon$ for $i < k$ (minimality of k) $\therefore t_{k-1} \in Q$ & $t_k \in Q'$

$\Rightarrow \boxed{s_k = \epsilon}$

For some $q_{k-1} \in \delta'(q_{k-1}, \varepsilon)$, keeping in mind δ', there must be some $a \in \Sigma$ such that $\delta(q_{k-1}, a) = q_k$. Further we have $t_{i+1} \in \delta'(t_i, s_i) = \{\delta(t_i, s_i)\}$ and thus $t_{i+1} = \delta(t_i, s_i)$ for $0 \leq i < k-1$. |||ly for $i \geq k$, we have $t_i \in Q'$. say $(t_i) = (t_i)'$ and then $'t_{i+1} = \delta('t_i, s_{i+1})$.

Let,

$s' = s_0 \ldots s_{k+1}\, a\, s_k \ldots s_n \in \Sigma^*$ and consider to

$t_0 \ldots t'_{k-1}\, t_k \ldots 't_n$. We have every $t_i \in Q$, $t_0 = q_0$, $'t_n \in F$

and

$$\delta(t_i, s_{i+1}) = t_{i+1} \quad 0 \leq i < k-1$$

$$\delta(t_{k-1}, a) = t_k$$

$$\delta('t_i, s_{i+1}) = 't_{i+1} \quad k \leq i < n$$

By the definition, D accepts s', so $s' \in L$. Further we have $s \in$ Delete (A) with $s = xyz$, $x = s_0 s_1 \ldots s_{k-1}$, $y = a$, $z = s_{k+1} \ldots s_n$

$\Rightarrow$ N accepts only strings in Delete (A)

$L(N) \supseteq$ Delete (A)

Let $s \in$ Delete (A) and let x, y, z be such that $x, z \in \Sigma^*$ and $y \in \Sigma$ and $xz = s$ and $s' = xyz \in A$. Then D accepts xyz, So we have $s' = s'_0 \ldots s'_n$ and a sequence of states $t_0 \ldots t_n$ where $t_n \in F$ and $\delta(t_i, s'_{i+1}) = t_{i+1}$ for $0 \leq i < n$.

Let $k = |x|$, so (s_k would be y) and $s = s'_0 \cdots s'_{k-1} \in s'_{k+1} \cdots s'_n$.

and let $q_0 \cdots q_n = t_0 \cdots t_{k-1} t'_k t'_{k+1} \cdots t'_n$. We have

q_0 = start state of Q and $q_n \in F'$ so q_n = final state of F'.

Now, for each $0 \leq i < k-1$. $q_{i+1} = t_{i+1} \in \{t_{i+1}\} = \delta'(t_i, s_i)$

for $i = k-1$

$$q_{i+1} = t'_k \in \{\delta(t_i, a)' \mid a \in \Sigma\} = \delta'(t_i, \varepsilon) = \delta'(q_i, s_{i+1})$$

Now since,

$$\delta(t_i, y) = t_k$$

and for $i \leq k < n$

$$q_{i+1} = t'_{i+1} \in \{t'_{i+1}\} = \delta'(t_i', s_i) = \delta'(q_i, s_i)$$

Thus by defination, N accepts s.

$\Rightarrow$ N accepts every string of Delete(A).

Hence Delete(A) is regular if A is regular

Hence Proved.

Que 4:-

(a) Consider the language of all binary strings with twice as many 0's and 1's. Give a CFG and a PDA for the language.

Ans: Considering the CFG

$G = (V, T, S, P)$ ⇒ It will generate the required language.

$$\Rightarrow G = \{ \underbrace{\{0,1\}}_{\text{Set of Variables}}, \underbrace{\{0,1\}}_{\text{Terminals}}, \underbrace{\{S\}}_{\text{Start Symbol}}, \underbrace{P}_{\text{Production Rules}} \}.$$

P = Production Rules, which are defined as.

$S \rightarrow SS$

$S \rightarrow 1S00$

$S \rightarrow 00S1$ or $S \rightarrow SS \mid 1S00 \mid 00S1 \mid 0S1S0 \mid \varepsilon$

$S \rightarrow 0S1S0$

$S \rightarrow \varepsilon$

Now, every rule of the above CFG produces string with twice as many 0's as 1's. Therefor this is the required CFG.

PROOF

→ In our languge empty string is valid which can be derived by $S \to \varepsilon$

→ Defing a function func(string) = no. of zeros − (2 × no of ones)
ie func(string) = no of zeros − 2 × no. of ones (in that string)

So,
for all the strings in our language func(string) = 0.

PROVING BY INDUCTION

→ Assuming that for all the strings $|s| < n$ can be produced for some $n \geq 0$

→ Let $|s| = n$ be any string present in our language.

(i) If s can be written as a combination of 2 strings $s = ab$ such that func(a) = 0, then func(b) = 0 because func(a) + func(b) = 0 as s belongs in our language. Thus the strings can be derived using $\boxed{S \to SS}$.

(ii) Considering all possible proper nontrivial prefixes p of string s such that func(P) > 0. then they must begin with 00. Since func(s) = 0 and score of func($s_1, s_2 \dots s_{n-1}$) is negative if $s_n = 0$, where (n = length of s) thus s_n must be 1. Therefore this string could be written as $00s'1$ and can be generated using $\boxed{S \to 00S1}$.

(iii) Also, if we consider proper non trivial suffixes p of s such that func (p) < 0, then those types can be made by $\boxed{S \rightarrow 1S00}$.

(iv) Considering some i such that func $(s_1 s_2 \ldots s_i) > 0$ and func $(s_1 s_2 \ldots s_{i+1}) \nless 0$ and no nontrivial prefix a exists such that func (a) = 0, then the following three inferences can be made.

1) $s_{i+1} = 1$

2) func $(s_1 s_2 \ldots s_i) = 1$

3) string s start with 0

Illy. for string $(s_{i+2} s_{i+3} \ldots s_n)$

1) func $(s_{i+2} s_{i+3} \ldots s_n)$ = func (w) − func (s_{i+1}) − func $(s_1 s_2 \ldots s_i)$

$$= 0 - (-2) - 1 = 1$$

2) string $s_{i+2} s_{i+3} \ldots s_n$ must end in 0.

Therefore; $s_2 s_3 \ldots s_i = s_{i+2} s_{i+3} \ldots s_{n-1} = 0$. Such strings can be easily derived using. $\boxed{S \rightarrow 0S1S0}$

⇒ Our CFG covers all the types of strings in our language and hence it is valid and therefore it is the required CFG.

PDA $\Rightarrow M = (Q, \Sigma, \Gamma, \delta, q_0, Z, f)$

$M = (\{q_0, q_1 q_2 q_3 q_4 q_5\}, \{0,1\}, \{0,1,z\}, \delta, q_0, Z, q_0)$

$\delta \Rightarrow$

$\delta(q_0, \lambda, \lambda) = (q_1, z)$

$\delta(q_1, \lambda, z) = (q_0, \lambda)$

$\delta(q_1, 0, \lambda) = (q_2, 0)$

$\delta(q, 1, \lambda) = (q_5, 0)$

$\delta(q_2, \lambda, z) = (q_1, z)$

$\delta(q_2, 0, \lambda) = (q_2, 0)$

$\delta(q_2, 1, 0) = (q_4, \lambda)$

$\delta(q_3, \lambda, z) = (q_1, z)$

$\delta(q_3, 0, 0) = (q_3, \lambda)$

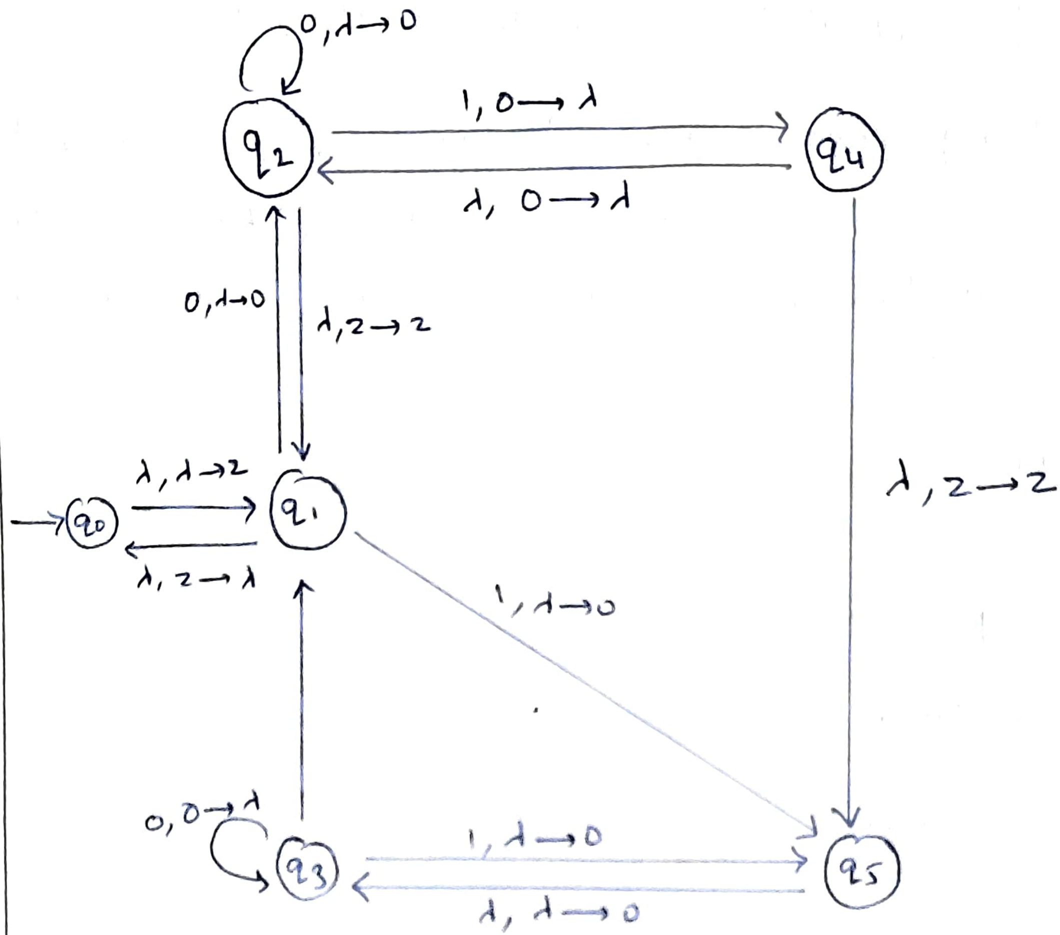
$\delta(q_3, 1, \lambda) = (q_5, 0)$

$\delta(q_4, \lambda, 0) = (q_2, \lambda)$

$\delta(q_4, \lambda, z) = (q_5, z)$

$\delta(q_5, \lambda, \lambda) = (q_3, 0)$

Required PDA



The above PDA accepts a string if it reaches q_0 with an empty stack or with the start symbol z at q_1.

(b) Prove that the following language is context free.

$L = \{s_1 s_2 \ldots s_n t_1 t_2 \ldots t_n \mid s_i \in L_1, t_i \in L_2, n \in N\}$

L_1, L_2 ~~are~~ context free languages.

Ans: Assuming the grammar of $L_1 = \{V_1, T, S_1, P_1\}$ and the grammar of $L_2 = \{V_2, T, S_2, P_2\}$

Considering a CFG given by $\{V, T, S, P\}$

$V = V_1 \cup V_2 \cup \{S\}$ $\quad T = T_1 \cup T_2$

$S \rightarrow$ new start state $\quad P = P_1 \cup P_2 \cup \{S \rightarrow S_1 S S_2 \mid \epsilon\}$

In this there is one new rule that is introduced that is

$S \rightarrow S_1 S S_2 \mid \varepsilon$

S_2 derives a string in L_2

S_1 derives a string in L_1

→ Due to the structure the order S_1 and then S_2 is maintained in any subsequent productions.

→ Since this is the only source of production in $S_1 S S_2$ then the no. of strings from each languages are also equal to some $n \in N$

→ This rule will either give an empty string ϵ or $S_1 S S_2$. Therefore the strings produced by the CFG are exactly those contained in the given language

→ As $(S \rightarrow S_1 S S_2 \mid \varepsilon)$ is correct and ~~both~~ rules for S_1 and S_2 are already context free it follows ~~that~~ all the rules of CFG

This is also a CFG

— x — x — x — x — x — x — x — x — x —