1901CS63

CS303 MIDSEM TANISHQ MALU

Q1

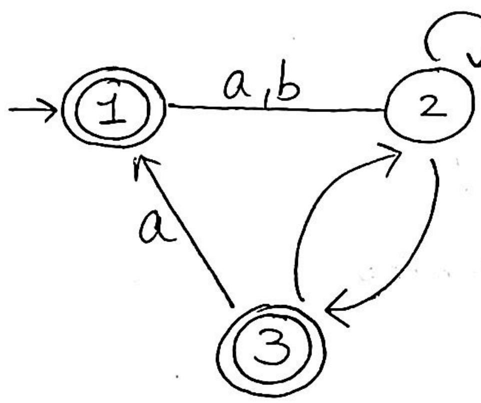
(a) $\Sigma = \{0,1\}$, $L =$ even no. of 0_s ~~and~~ or exactly two 1s

Sol $1^*(01^*01^*) + (0^*10^*10^*)$

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

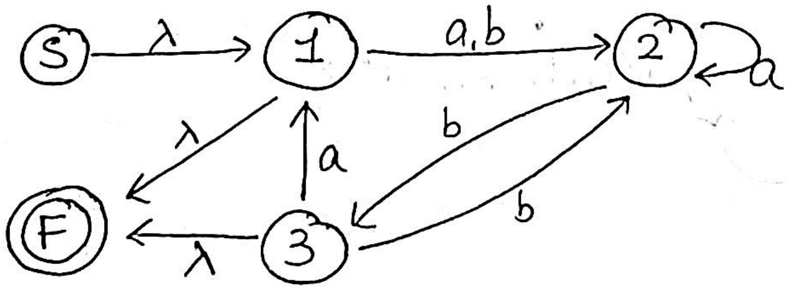
Sol $0 + \varepsilon$

(c) DFA ⇒



Converting our DFA to GNFA to find the regular expression.

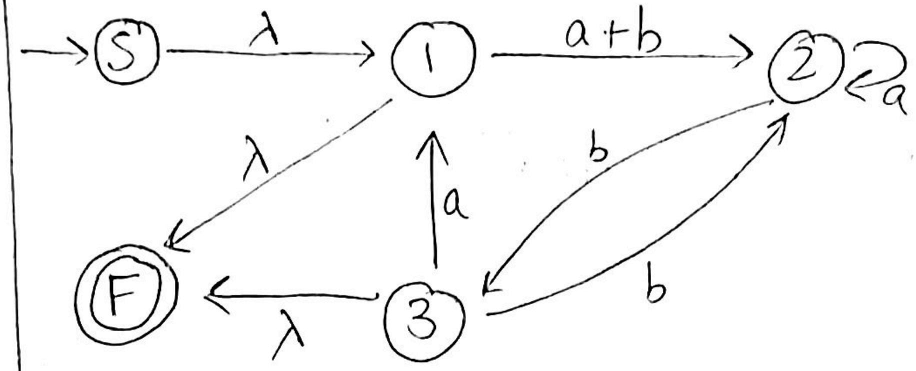
1.) Adding new start state and new final state. Making original final states as nonfinal. Adding required ε transition

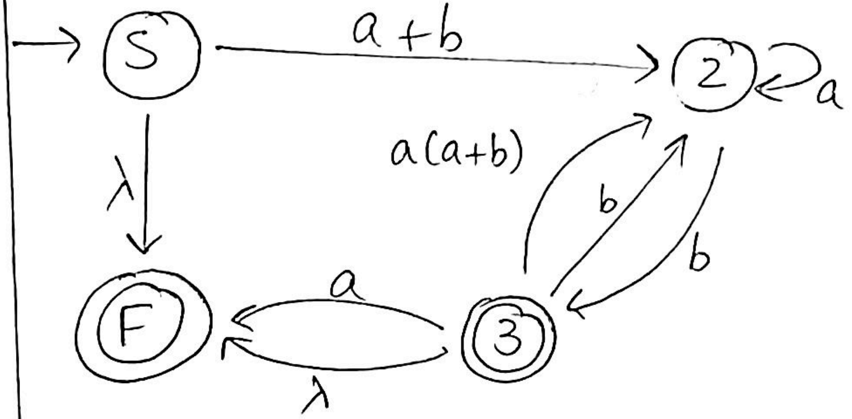


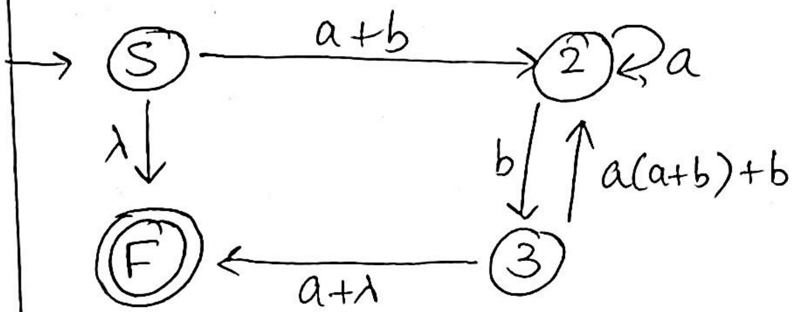
2.) Reducing states ⇒ (1.) Union : $A \xrightarrow{0,1} B \Rightarrow A \xrightarrow{0+1} B$

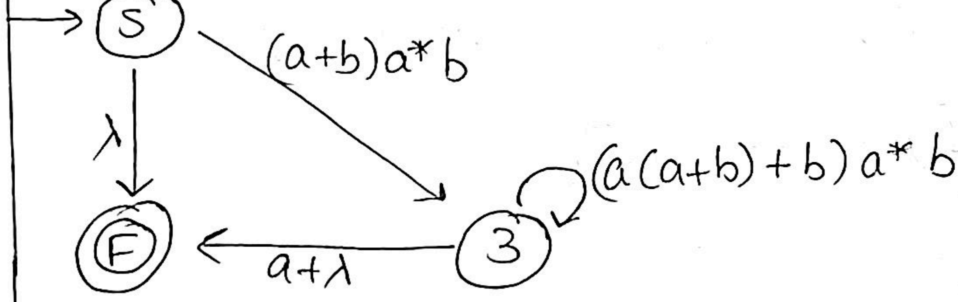
(2.) Internal state : $A \xrightarrow{x} R \xrightarrow{y} B \Rightarrow A \xrightarrow{xy} B$

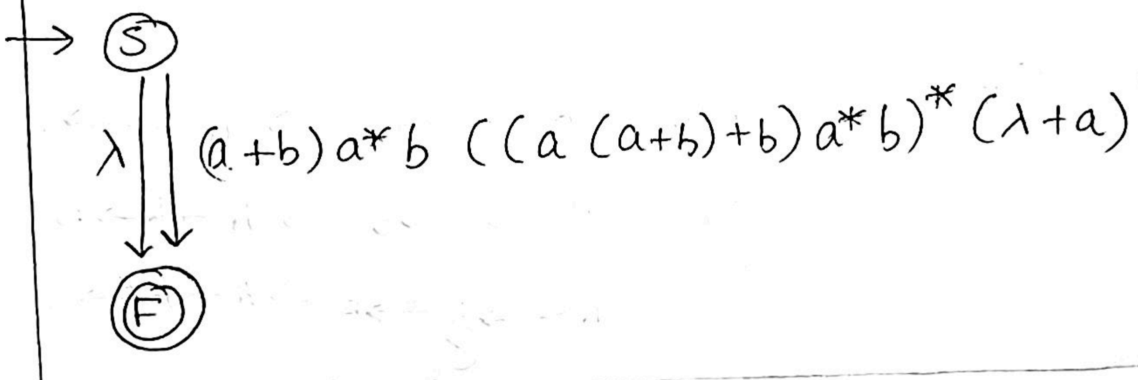
$A \xrightarrow{x} \underset{\circlearrowleft y}{R} \xrightarrow{z} B \Rightarrow A \xrightarrow{xy^*z} B$


S
λ
1
a+b
2
a
λ
a
b
b
F
λ
3


S
a+b
2
a
λ
a(a+b)
b
b
F
a
λ
3


S
a+b
2
a
λ
b
a(a+b)+b
F
a+λ
3


S
(a+b)a* b
λ
(a(a+b) + b) a* b
F
a+λ
3


S
λ
(a +b) a* b ((a (a+b)+b) a* b)* (λ+a)
F

Final regex

$\Rightarrow \lambda + ((a+b)a^*b)((a(a+b)+b)a^*b)^*(\lambda+a)$

(d) $\Sigma = \{a,b\}$ L= odd no. of as and ends with ab

Sol $b^*(ab^*ab^*)^*ab$

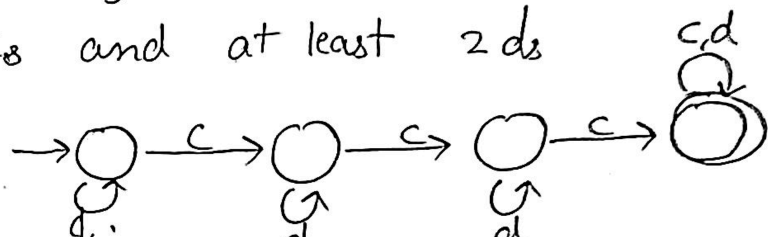
(e) $\Sigma = \{0,1\}$ L= strings except 11 and 111

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

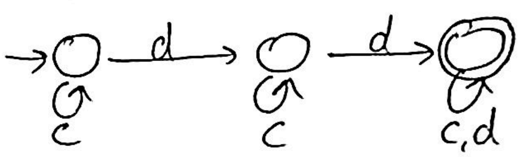
Q2 Draw DFA for $\Sigma = \{c,d\}$

(a) w|w has at least 3 cs and at least 2 ds

Sol DFA for at least 3 cs

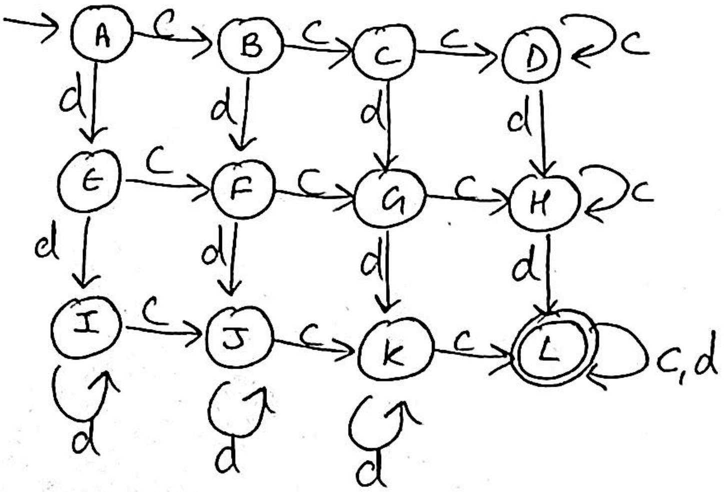


DFA for at least 2 ds



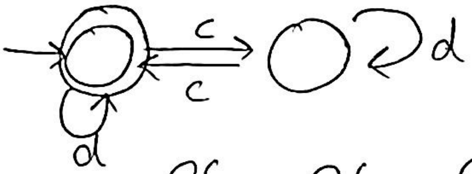
On taking intersection

DFA ⇒

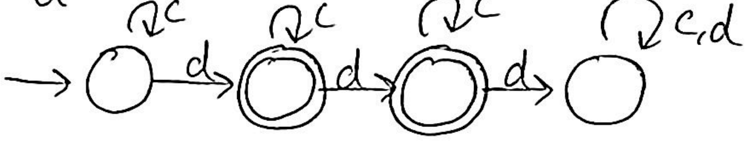


(b) w|w has even no. of cs and one or two ds

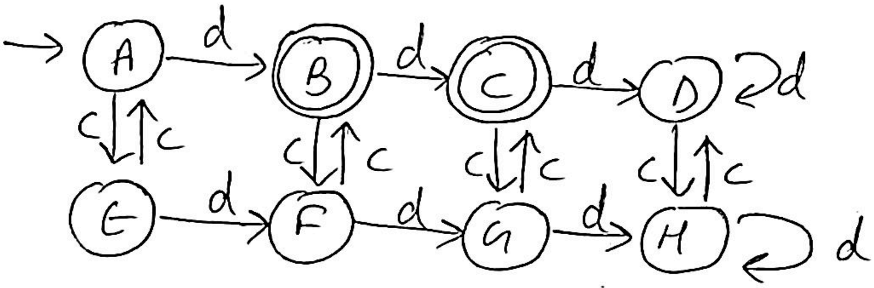
DFA for even cs



DFA for one or two d

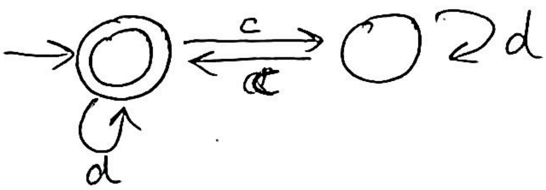


Intersection ⇒

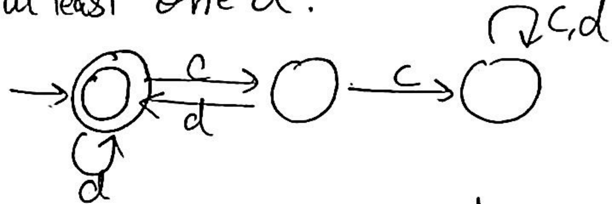


(C) w|w has an even no. of cs and each c is followed by at least one d.

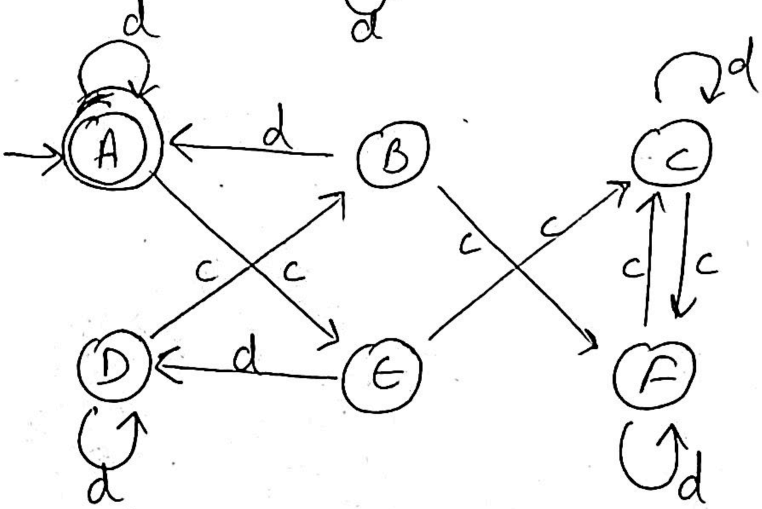
DFA for even cs -



DFA for each c followed by at least one d.

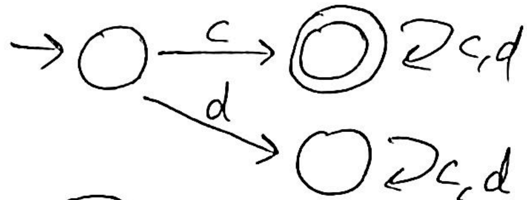


Intersection ⇒

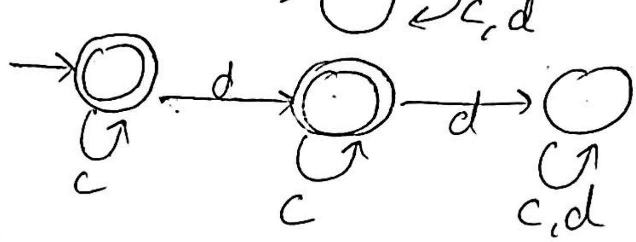


(d) w|w start with an c and has at most one d

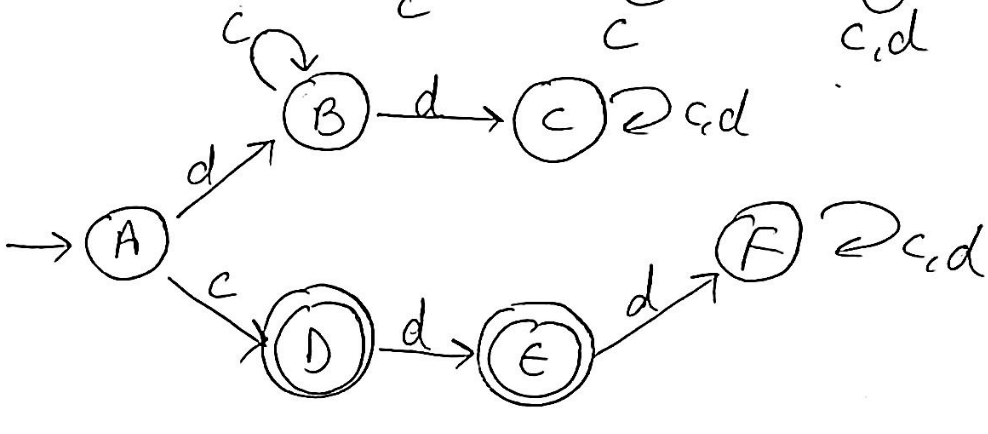
DFA for start with c



DFA for at most one d



Intersection:



Q4

(a) Consider a language of all binary strings with twice as many 0s as 1s.

Sol Let us consider a CFG $G = (V, T, S, P)$ which can generate the required language. Therefore, this

$G = \{\{0,1,S\}, \{0,1\}, \{S\}, P\}$

set of variables — Terminals — Start symbol — Production rules

where P = production rules are defined below –

$S \rightarrow SS$

$S \rightarrow 1S00$

$S \rightarrow 00S1$

$S \rightarrow 0S1S0$

$S \rightarrow \varepsilon$

$\boxed{\text{OR}}$

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

Every rule of this CFG produces string with twice as many 0s as 1s. Thus this is our required CFG.

Proof

- The empty string is valid in our language and can be derived by $S \rightarrow \varepsilon$.
- Let us now define a function score (string) which is equal to no. of zeroes − (2×no. of ones) i.e

score (string) = no. of zeroes − 2× no. of ones (in that string)

Thus for all strings in our language $\underline{score(string) = 0}$

Let us prove by induction

- Let's assume that for all strings $|s| < n$ can be produced for some $n \geq 0$
- Let $|s| = n$ be any string present in our language.

(1.) If s can be written as a combination of 2 strings $s = ab$ such that $\underline{score(a) = 0}$, then $score(b) = 0$ because $score(a) + score(b) = 0$ as s belongs in our language. Thus, such strings can be derived using

$\boxed{S \to SS}$

(2.) Lets consider all possible proper nontrivial prefixes p of a string s such that $score(p) > 0$ then they must begin with 00. Since $score(s) = 0$, and score of score $(s_1, s_2 \ldots s_{n-1})$ 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

$\boxed{S \to 00S1}$

(3) Similarly if we consider proper nontrivial suffixes p of s such that $score(p) < 0$. then these types can be made by $\boxed{S \to 1S00}$

(4) Let us consider some i such that $score(s_1 s_2 \ldots s_i) > 0$ and $score(s_1 s_2 \ldots s_{i+1}) < 0$ and no nontrivial

prefix a exists such that score(a) = 0, then 3 inferences can be made

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

(ii) score $(s_1 s_2 \dots s_i) = 1$

(iii) string s start with 0.

similarly for string $(s_{i+2}\, s_{i+3} \cdots s_n)$

(i) score $(s_{i+2}\, s_{i+3} \cdots s_n)$ = score (w) − score (s_{i+1}) − score $(s_1 s_2 \cdots s_i)$

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

(ii) string $s_{i+2}\, s_{i+3} \cdots s_n$ must end in 0.

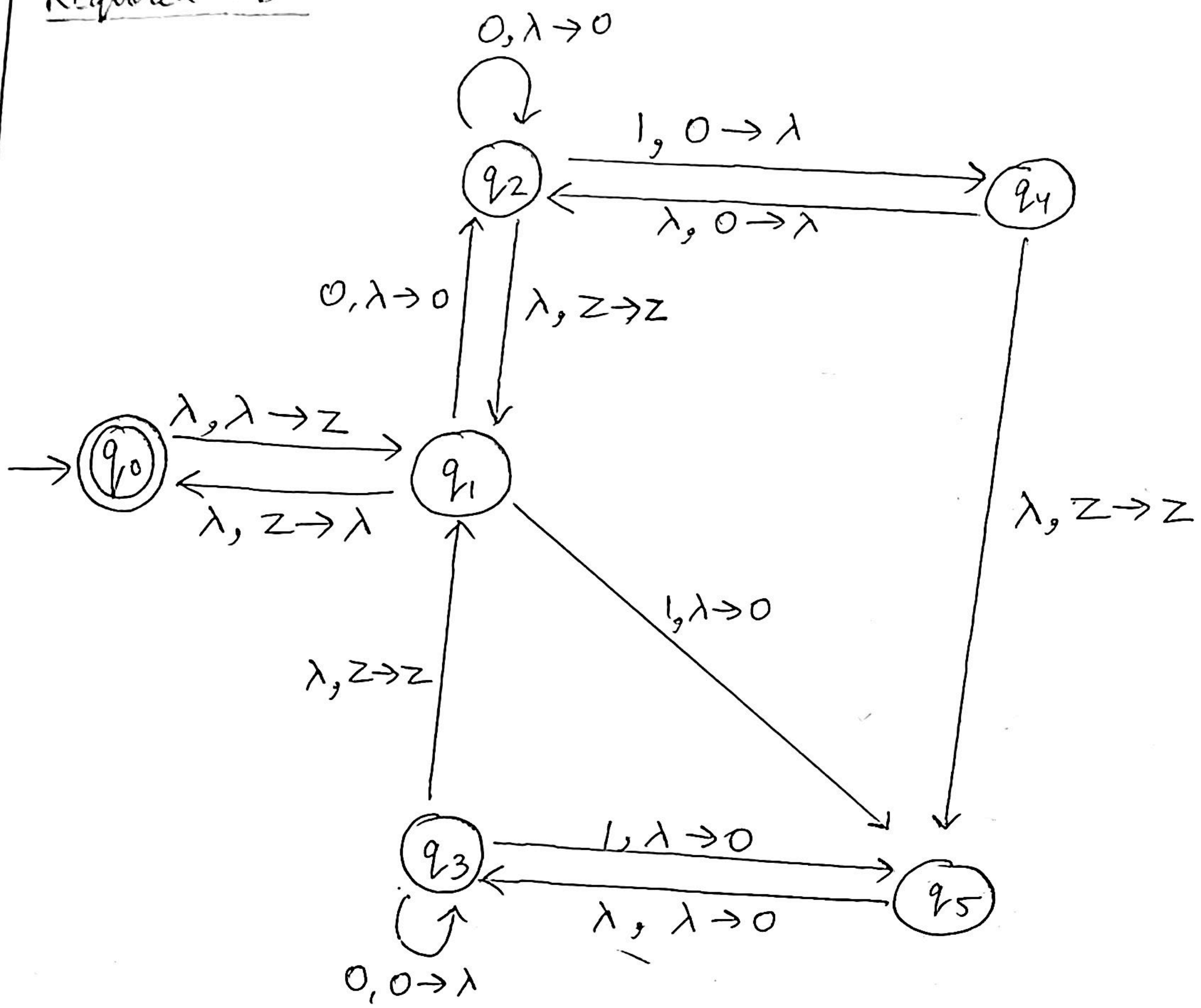
Hence $s_2 s_3 \cdots s_i = s_{i+2}\, s_{i+3} \cdots s_{n-1} = 0$

Such strings can be easily derived using

$\boxed{S \rightarrow 0\,S\,1\,S\,0}$

Thus our CFG covers all types of strings in our language and hence is valid.

Required PDA



This PDA accepts a string if it reaches q_0 ~~(or q_f)~~ with an empty stack or with the start symbol z at q_1.

(b) To prove that following language is context free:

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

where L_1 and L_2.

Let the grammar of L_1 be $\{ V_1, T, S_1, P_1 \}$ and that of L_2 be $\{ V_2, T, S_2, P_2 \}$. We can rename the variables ~~of~~ differently so that no two variables name in L_1 and L_2 are same.

Let us consider a CFG given by $\{V, T, S, P\}$
where $V = V_1 \cup V_2 \cup \{S\}$ where S is new start state

$T = T_1 \cup T_2$

S

$P = P_1 \cup P_2 \cup \{ S \rightarrow S_1 S S_2 \mid \varepsilon \}$

Thus only one new rule is introduced that is

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

S_1 derives a string in L_1

S_2 derives a string in L_2

→ also the order S_1 then S_2 is maintained in any subsequent productions due to its structure.

→ 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 us an empty string ε or $S_1 S S_2$. Thus the strings produced by this CFG are exactly those contained in our given language.

→ It follows all the rule of CFG as $(S \rightarrow S_1 S S_2 \mid \varepsilon)$ is correct and rules for S_1 and S_2 are already Context-free. Hence this is also a CFG.