Automata Theory - Assignment-1

NAGAMANOHAR
2021101128

SECTION-1:

(1)
(a) Spurious States :- (4), (5)

(1) Dead States :- None
(2) Unreachable States :- (4), (5)

(b) Spurious States :- (3), (5)

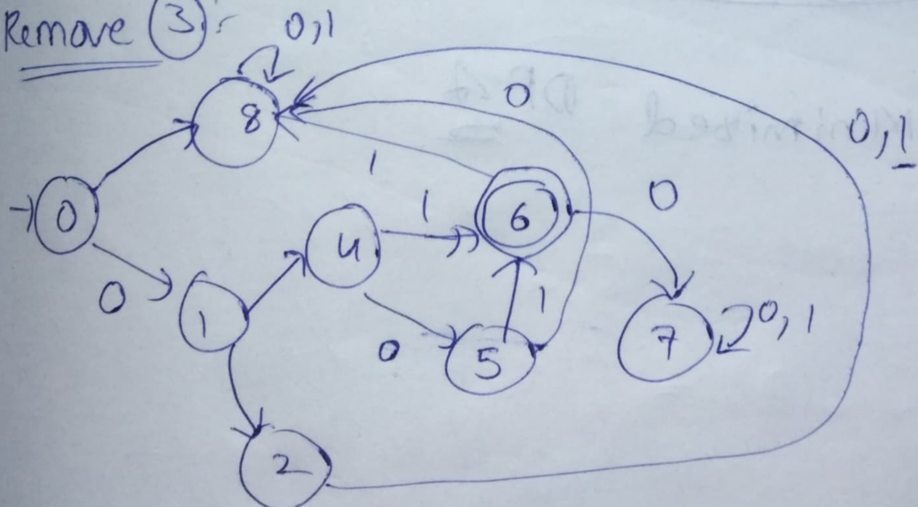
(1) Dead States :- (3), (5)
(2) Unreachable States :- None

(2)
There are (4) dead states in the given DFA

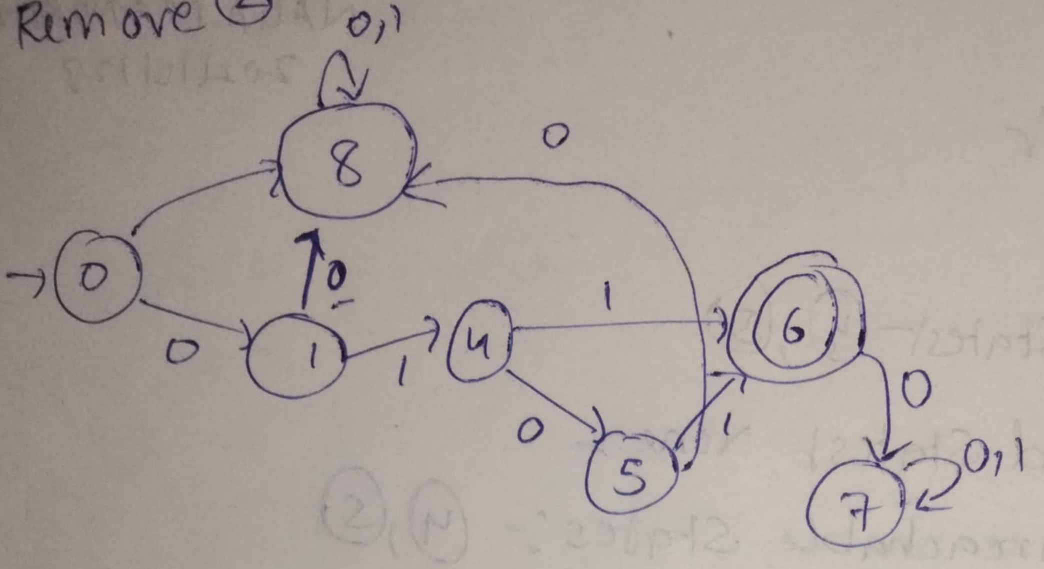
- we need only one of it.

Dead States are :- 2, 3, 7, 8

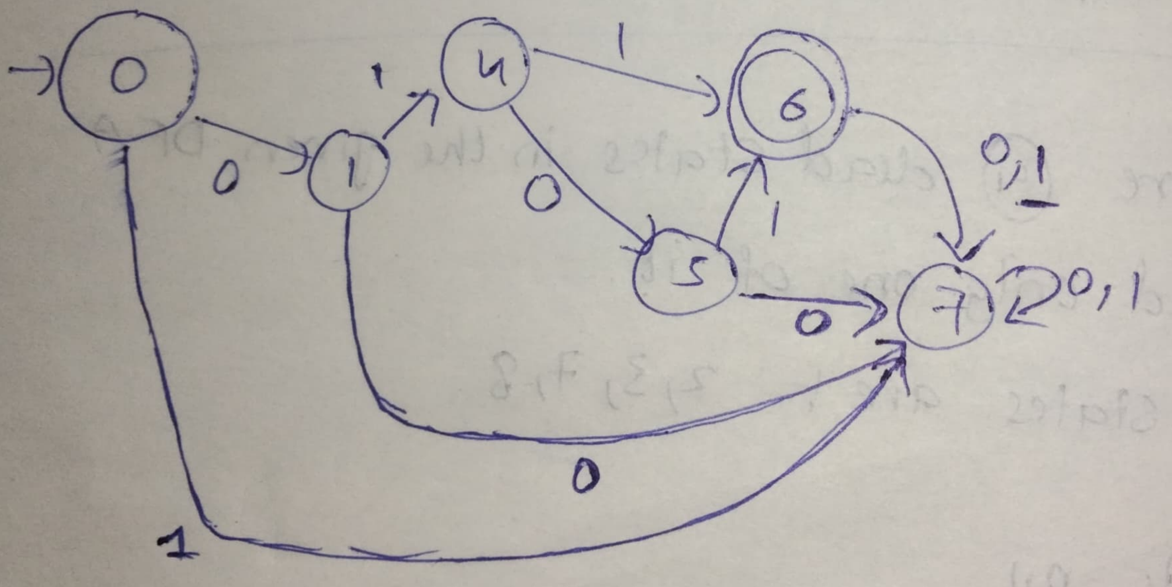
Remove (3):-



Remove 2



Remove 8 and shift all incoming arrows to 7



- This is Minimized DFA

(3)

Given a Language L with Σ

$$Pre(L) = \{x \in \Sigma^* \mid \exists y \in \Sigma^* \text{ s/t } xy \in \Sigma^*\}$$

RTPL If L is regular then $Pre(L)$ is regular.

We can make an Automaton that recognizes $pre(L)$.

As L is Regular $\Rightarrow L$ has DFA M

Now by making all states that reach final state in M as Final states the DFA will recognize $pre(L)$

So let $A = (Q, \Sigma, \delta, q_0, F)$ - be DFA that accepts L

Now we get New DFA
$$D = (Q, \Sigma, \delta, q_0, F_1)$$

Now $$F_1 = \{q \in Q : \exists y \in \Sigma^*, \delta^*(q, y) \in F\}$$

let $x \in \Sigma^*$, $x \in L(D) \Leftrightarrow \delta^*(q_0, x) \in F_1$

$\Leftrightarrow \exists y : \delta^*(\delta^*(q_0, x), y) \in F$

$\Leftrightarrow \exists y : \delta^*(q_0, xy) \in F$

$\Leftrightarrow \exists xy \in L$

$\Leftrightarrow x \in L$

$\therefore pre(L)$ is regular

(4)

Given,

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

RTP: $Dropout^*(A)$ is regular if A is regular.

we prove this by showing that we can build an NFA for $Dropout^*(A)$.

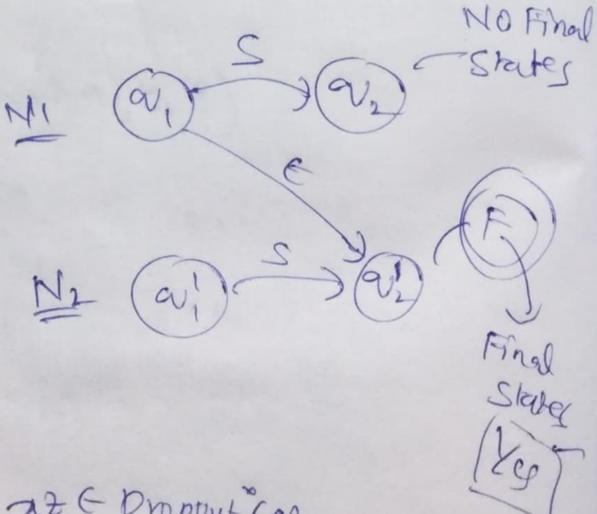
A is regular ⇒ ∃ a NFA for A

Now make 2 copies of that NFA

Call — N_1, N_2

Now let there be a transition from state q_1 to q_2 in N_1 through symbol s

Now to skip symbol s we take q_1 in N_1 to copy of q_2 in N_2 through (let q_2') ϵ transition



So we do this for all possible transitions

Hence if $xyz \in A$

y is skipped and $xz \in Dropout^*(A)$

Here don't take any final states in N_1

as they will go to N_2 to form xz

and will only be accepted in N_2.

Hence we built a NFA that recognizes

Dropout(A)

$\Rightarrow$ Dropout(A) is regular

(5)

Regular Expression:-

$(a+b+c+d+e+f+g+h+i+j+k+l+m+n+o+p+q+r+s+t+u+v+w+x+y+z)^*.$(@.iiit.ac.in + @research.iiit.ac.in + @students.iiit.ac.in)

(or)

let

$\Sigma = \{a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z\}$

Then,

Regular Expression:-

$\Sigma^*.$(@iiit.acin + @research.iiit.acin + @students.iiit.acin)

(6)

(a)

Algorithm to Convert Regular Expression to right linear Grammar

(1) Make the NFA for Given Regular Expression

(2) Now let P, Q be 2 states and there can be transitions like

$P \xrightarrow{\epsilon} Q$ in NFA

$P \xrightarrow{a} Q$

Now we can convert this to right linear Grammar as $P \to Q$, $P \to aQ$ ⇒ production Rules

- Here the transition should be outgoing transition

(3) we write production Rule of START state at the beginning.

- Mark a state Q as Final state by giving a rule $\boxed{Q \to \epsilon}$

(b) Algorithm to convert right linear grammar to a left linear grammar

① Given the right linear grammar, as a Regular grammar is analogous to DFA/NFA, construct the Finite Automata corresponding to the Right-linear Grammar.

② Now Interchange/Swap START state with the final states

③ Then Reverse all the arrows (transition arrows)

- Incoming transition → becomes outgoing
- Outgoing transition → becomes incoming

④ Now make the left linear Grammar from Finite Automata by using the ~~rule~~ incoming transitions

let P, Q be states in AUTOMATA s/t

$P \xrightarrow{a} Q$ then

left-linear grammar ⇒ $\boxed{Q \to Pa}$

If P is start state then use

$\boxed{Q \to a}$

Thus we build ~~left~~ LG from Finite Automata

$\boxed{\Rightarrow \text{RLG} \to \text{LLG}}$

7(a)

Given,

$A = \{ bits(n) \mid len(bits(n))$ is prime, $n \in N \}$

RTP: A is NOT Regular

By pumping lemma:

Assume A is regular

Let p be the pumping length, and

let $s = string(bits(n)) \in A$ of size $\leq p$

$\Rightarrow len(s) \leq p$

Now split $s = xyz \in A \Rightarrow xy^i z \in A \; \forall i \geq 0$

Now $len(s)$ is prime

and let length of part of s x be $|x|$

length of y be $|y|$

length of z be $|z|$

$\Rightarrow len(s) = |x| + |y| + |z|$

Now (xy^iz) let $\boxed{i = |x| + |z|}$

$\Rightarrow len(xy^iz) = |x| + (|x|+|z|)|y| + |z|$

$= (|x| + |z|) + (|x|+|z|)|y|$

$= (|x|+|z|)(1 + |y|)$

$=$ NOT a prime

$\Rightarrow xy^{|x|+|z|}z \notin A$

This a Contradiction!

Thus a contradiction is unavoidable if we make the assumption that A is regular.

Hence, A is NOT Regular

7 (b)

Given,

$L = \{a^{n!} \mid n \geq 0\}$

RTP: L is NOT Regular

Assume that L is Regular

Let p be the Pumping length (by pumping lemma) for L.

Now let

s = string in L with length $\geq p$

Now, $\Rightarrow s = xyz$, and $\forall i \geq 0$ $xy^iz \in L$ should hold True

- The string y be a segment of a's of size k

Now let ~~p = k~~ then

$= x.y.y.y.y\,z$

$= a^{n! + 3k}$

and let k = 1 $\Rightarrow a^{n!+3} \notin L$

and this holds true for any $k \geq 1$

Hence, this a contradiction!!

$n!+3$ is NOT a factorial of any number

Thus a contradiction is unavoidable if we make the assumption that L is regular,

Hence, L is NOT Regular

(8) $L = \{a,b\}^* - \{palindromes\}$

Yes,

L is Context Free.

Because we can write a CFG for L

and if CFG can be written then

PDA can be made

$\Rightarrow$ L is Context Free

For palindromes

CFG $\Rightarrow$ $S \to aSa \mid bSb \mid a \mid b \mid \epsilon$

All $\{a,b\}^*$ CFG

$\Rightarrow$ $S \to aS \mid bS \mid \epsilon$

Now to NOT get any palindromes we add

2 extra Transitions as aQb, bQa

Sothat No palindrome is formed

Hence, CFG will be

$P \to aPa \mid bPb \mid bQa \mid aQb$

$Q \to aQ \mid bQ \mid \epsilon$ $\quad - (a+b)^*$

Now <u>CGF to CNF</u>

① Add new start-variable (S')

$S' \to P$

$P \to aPa \mid bPb \mid bQa \mid aQb$

$Q \to aQ \mid bQ \mid \epsilon$

(2) Remove $A \to \epsilon$ transitions

(1) Remove $Q \to \epsilon$

$S' \to P$
$P \to aPa \mid bPb \mid bQa \mid aQb \mid ba \mid ab$
$Q \to aQ \mid bQ \mid a \mid b$

(3) Remove single variable rules $A \to B$

i) Remove $S' \to P$

$\Rightarrow$ $S' \to aPa \mid bPb \mid bQa \mid aQb \mid ba \mid ab$
$P \to aPa \mid bPb \mid bQa \mid aQb \mid ba \mid ab$
$Q \to aQ \mid bQ \mid a \mid b \mid$

(4) Now, Remove long string (variables, Terminals) by adding rules

$S' \to aU \mid bV \mid bW \mid aX \mid ba \mid ab$
$P \to aU \mid bV \mid bW \mid aX \mid ba \mid ab$
$Q \to aQ \mid bQ \mid a \mid b$

$U \to Pa$
$V \to Pb$
$W \to Qa$
$X \to Qb$

Now add ~~$T \to aQ$~~ ~~$R \to bQ$~~ $A \to a$
$B \to b$

$S' \to AU \mid BV \mid BW \mid AX \mid BA \mid AB$
$P \to AU \mid BV \mid BW \mid AX \mid BA \mid AB$
$Q \to AQ \mid BQ \mid A \mid B$

$U \to Pa$ $\quad X \to Qb$
$V \to Pb$ $\quad A \to a$
$W \to Qa$ $\quad B \to b$

Hence
var $\to$ var var
var $\to$ ter

Hence, CGF $\to$ CNF is made

(9) $L = \{a^n b^n \cup b^n a^n \mid n \geq 0\}$

we have to find the CFG for

complement of L

$\Rightarrow$ All strings of $\{a, b\}$ − $a^n b^n \cup b^n a^n$

$P \rightarrow aQb \mid bRa \mid aSa \mid bSb \mid a \mid b$

$Q \rightarrow aQb \mid bSa \mid aSa \mid bSb \mid a \mid b$

$R \rightarrow bRa \mid bSa \mid aSa \mid bSb \mid a \mid b$

$S \rightarrow aS \mid bS \mid \epsilon$ — $(a+b)^*$

This is the CFG for the L complement.

Here we make $a^n b^n$, $b^n a^n$ with Q, R

but before reaching ~~Terminal symbols~~ Final Derivation

we go to S (or) add only single a/b

so that No string will be of the

form $a^n b^n$ or $b^n a^n$.

Here S represents $(a+b)^*$

(10)

Given languages A, B over Σ

$$min(A,B) = \{ w \mid w = a_1 b_1 a_2 b_2 \ldots a_k b_k$$

where $a_i, b_i \in \Sigma \ \forall i$,

& $a_1 a_2 \ldots a_k \in A$
$b_1 b_2 \ldots b_k \in B \}$

RTP: min(A,B) is regular when A, B are regular

Now consider
$D_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ — DFA for A
$D_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ — DFA for B

we shall prove by constructing a DFA D that recognizes min(A,B)
$D = (Q, \Sigma, \delta, q, F)$
using D_A & D_B

Here DFA D — will be running D_A and D_B alternatively by switching between them thus forming min(A,B)

Here we need to know:
① The current states of D_A and D_B
② The next state it is going to switch to, to be matched (A/B)

- By the end, if D_A & D_B are in Final States then the string is accepted by D. else No.

Now we Define Terms in D :-

(i) $Q = Q_A \times Q_B \times \{A, B\}$

- All possible current states of A, B are $(Q_A \times Q_B)$, which DFA to match with (A/B)

(ii) Start state: $q = (q_A, q_B, A)$

- D starts from q_A of D_A and q_B of D_B then points to D_A for reading next character. eg: $a_1 b_1 a_2 \Rightarrow a_2 \in D_A$

(iii) Final state: $F = F_A \times F_B \times \{A\}$

- D Accepts only if D_A and D_B Accept, also as string ends with $b_k \in B$, the next character read should be in A

(iv) δ (Transition func) :-

① $\delta((x, y, A), a) = (\delta_A(x, a), y, B)$

x – current state of D_A

y – current state of D_B

a – next character read

A – next character read will be in D_A

W.K.T $\delta_A(x, a)$ = State after transition from x with input a.

- State of D_B Doesn't change (y)

$\Rightarrow \delta((x, y, A), a) = (\delta_A(x, a), y, B)$ will be in next character ~~to be~~ read ~~from~~ D_B

similarly

(2) $\delta((x, y, B), b) = (x, \delta_B(y, b), A)$.

Hence for D_A, D_B $\exists$ D s/t

if A, B are regular then min(A, B) is regular by forming Finite Automata D

Hence, proved

(11) Given

a Language

$L = \{a^i b^j c^k \mid i,j,k \geq 0,\ i=j \text{ (or) } j=k\}$

Find: PDA for L.

There are 2 cases

$i=j \Rightarrow n \Rightarrow a^n b^n c^k$ - ①

$j=k=n \Rightarrow a^i b^n c^n$ - ②

Now combining both we get the conditions - following PDA.