Q.2 a. Prove by mathematical induction $n^4 - 4n^2$ is divisible by 3 for $n \geq 0$.

Answer:

Basic step: For n = 0, $n^3 - n = 0$ which is divisible by 3.
Induction hypothesis: Let $p(n) = n^3 - n$ is divisible by 3.
Induction step: Let us prove this for (n+1) also.

$$\begin{aligned}\text{Then } p(n+1) &= (n+1)^3 - (n+1)\\ &= (n^3 + 3n^2 + 3n + 1) - (n+1)\\ &= n^3 + 3n^2 + 3n - n\\ &= (n^3 - n) + 3(n^2 + n)\end{aligned}$$

Now $(n^3 - n)$ is divisible by 3 as p(n) is true by induction hypothesis. Also $3(n^2 + n)$ is a multiple of 3 and hence divisible by 3.
Thus $p(n) = n^3 - n$ is divisible by 3 for all $n \geq 0$.

b. What is the need to study Automata Theory in computer science?

Answer: Page Number 6, 7 of Text Book

Q.3 a. Minimize the following DFA having state q_5 as final state:

Present State	Next State	
	Input 0	Input 1
q_0	q_1	q_2
q_1	q_3	q_4
q_2	q_5	q_6
q_3	q_3	q_4
q_4	q_5	q_6
q_5	q_3	q_4
q_6	q_5	q_6

Answer:

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$, since q_5 is the final state.
$\Pi_0 = (q_5, Q - q_5)$ is the 0-equivalence class.

Construction of 1-equivalence class:

As $\delta(q_0, 0) = q_1$ and $\delta(q_0, 1) = q_2$, and both q_1 and q_2 are non final states.
$\delta(q_1, 0) = q_3$ and $\delta(q_1, 1) = q_4$, and both q_3 and q_4 are non final states.
Thus $q_0 \equiv q_1$
Proceeding in the same way, $q_0 \equiv q_3$
Thus $q_0 \equiv q_1 \equiv q_3$
Again $\delta(q_2, 0) = q_5$ and $\delta(q_4, 0) = q_5$ and $\delta(q_4, 1) = q_6$, and $\delta(q_2, 1) = q_6$, Hence $q_2 \equiv q_4$ and also $q_4 \equiv q_6$
Thus $\Pi_1 = [\{q_5\}, \{q_0, q_1, q_3\}, \{q_2, q_4, q_6\}]$ is the 1-equivalence class.
Similarly $\Pi_2 = [\{q_5\}, \{q_0, q_1, q_3\}, \{q_2, q_4, q_6\}]$ is the 2-equivalence class also.
Hence we have constructed the minimized sate automata as:
$Q = \{[q_0, q_1, q_3], [q_2, q_4, q_6], [q_5]\}$ and transition table is given below:

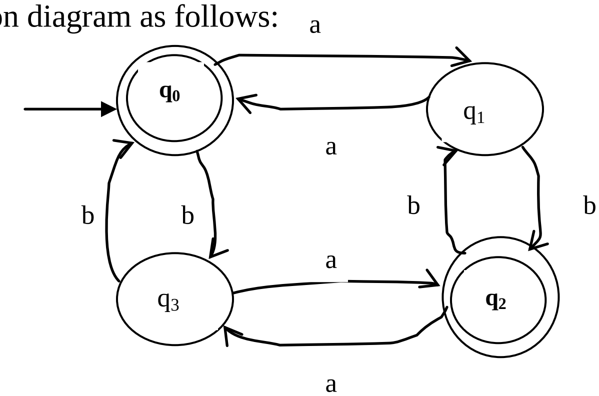
State	Input	
	0	1
$[q_0, q_1, q_3]$	$[q_0, q_1, q_3]$	$[q_2, q_4, q_6]$
$[q_2, q_4, q_6]$	$[q_5]$	$[q_2, q_4, q_6]$

[q_5]	[q_0, q_1, q_3]	[q_2, q_4, q_6]

b. Design a finite automata for the language L = {w| w is of even length and $w \in (a, b)^*$}.

Answer:

We design the transition diagram as follows:



Where $\mathbf{q_0}$ and $\mathbf{q_2}$ are the two final states.

Hence the finite machine is defined as:

$Q = \{q_0, q_1, q_2, q_3\}$

$\Sigma = (a, b)$

δ is the transition function given above.

q_0 is the initial state.

q_0 and q_2 are the two final states.

Q.4 a. Let $V_N = \{S, B\}$, $V_T = \{a, b\}$, $P = \{S \to aBa, B \to aBa, B \to b\}$. Find the language L(G) generated by the given grammar.

Answer:

From the given productions we have:

$S \to aBa \to aba$, hence $aba \in L(G)$.

$S \to aBa \to aaBaa \to aaaBaaa \to \ldots\ldots \to aa\ldots aBaa\ldots a \to a^nba^n \in L(G)$.

Hence { aba, aabaa, a^3ba^3,…., a^nba^n} = $\{a^nba^n \mid n \geq 1\} \subseteq L(G)$ ………(1)

To show that $L(G) \subseteq \{a^nba^n \mid n \geq 1\}$, we start with $w \in L(G)$, the derivation of w starts with S. If $S \to aBa$ is applied first and then $B \to b$, we get w = aba. On the other hand if we apply $B \to aBa$ ((n-1) times) and then finally to terminate this we apply $B \to b$, then w will be of the form a^nba^n,

So $w \in$ { aba, aabaa, a^3ba^3,…., a^nba^n}. Hence $L(G) \subseteq \{a^nba^n \mid n \geq 1\}$ …….(2)

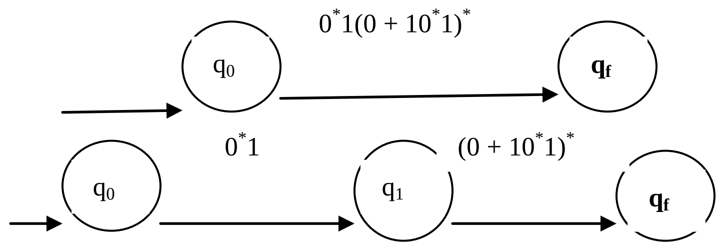
By (1) and (2) we have $L(G) = \{a^nba^n \mid n \geq 1\}$.

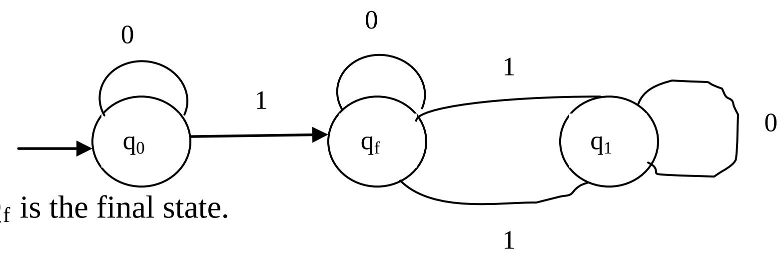
b. Obtain the NFA without epsilon transition corresponding to the following regular expression:

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

Answer:

Let us construct the NFA as follows:

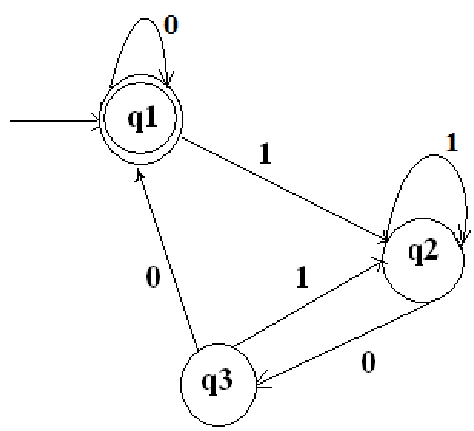




State Q_f is the final state.

Hence the NFA M = ({q_0, q_1, q_f}, {0, 1}, δ, {q_0}, {q_f}), the transition function δ is given above.

Q.5 a. Construct a regular expression corresponding to the state diagram given below



Answer:

There is only one initial state. Also, there are no null moves. The equations are:

$$q_1 = q_1 0 + q_3 0 + \wedge \quad …….(1)$$
$$q_2 = q_1 1 + q_2 1 + q_3 1 \quad …..(2)$$
$$q_3 = q_2 0 \quad ……(3)$$

So, $q_2 = q_1 1 + q_2 1 + (q_2 0)1 = q_1 1 + q_2(1 + 01)$ (put q_3 from eq. 3 to eq.2)

Applying Arden's theorem, which says if there is an equation in regular expressions P, Q and R as R = Q + R.P, then the solution is given by $R = Q.P^*$.

$$q_2 = q_1 1.(1 + 01)^*$$

Also we have

$q_1 = q_1 0 + q_3 0 + \wedge = q_1 0 + (q_2 0)0 + \wedge$ (by eq. (3))

Put the value of q_2 here which we have obtained above,

$q1 = q_1 0 + (q_1 1(1 + 01)^* 0)0 + \wedge$

$= q_1(0 + 1(1 + 01)^* 00) + \wedge$

Thus $q_1 = \wedge((0 + 1(1 + 01)^* 00)^* = (0 + 1(1 + 01)^* 00)^*$ (By Arden's Theorem)

As q_1 is the only final state, the regular expression corresponding to the given diagram is $(0 + 1(1 + 01)^* 00)^*$.

b. Consider the following productions representing regular grammar G,

$S \rightarrow aA \mid a$

$A \rightarrow aA \mid aB \mid a$

$B \rightarrow bB \mid c$

Find the regular expression corresponding to regular grammar G.

Answer:

Let us construct the language by the given productions:

$S \rightarrow a$ hence string $w = a \in L$

$S \rightarrow aA \rightarrow aaA \rightarrow aaaA \rightarrow \ldots\ldots \rightarrow aaa\ldots a^{n-1}A \rightarrow a^n \in L$.

$S \rightarrow aA \rightarrow aaA \rightarrow \ldots\ldots \rightarrow aaa\ldots a^{n-1}A \rightarrow a^nB \rightarrow a^nc \in L$.

$S \rightarrow aA \rightarrow aaA \rightarrow \ldots\ldots \rightarrow aaa\ldots a^{n-1}A \rightarrow a^nB \rightarrow a^nbB \rightarrow \ldots \rightarrow a^nb^m \in L$.

$S \rightarrow aA \rightarrow aaA \rightarrow \ldots\ldots \rightarrow aaa\ldots a^{n-1}A \rightarrow a^nB \rightarrow a^nbB \rightarrow \ldots \rightarrow a^nb^mB \rightarrow a^nb^mc \in L$.

Hence $L = \{a^nb^m \cup a^nb^mc \mid n \geq 1, m \geq 0\}$

It's regular expression can be written as $\mathbf{a^+b^* + a^+b^*c = a^+b^*(\varepsilon + c)}$.

Q.6 a. Construct a PDA to accept strings containing equal number of 0's and 1's by null store. Show the moves of the PDA for the input string '011001'.

Answer:

Let $M = \{(q_0, q_1), \Sigma = (0, 1), (a, b, Z_0), \delta, \{q_f\}, q_0, Z_0\}$ where Z_0 is the special stack symbol which says that stack is empty.

Now we write the push and pop operations as follows:

Push Operations:

1. $\delta(q_0, 0, Z_0) = (q_0, 0Z_0)$
2. $\delta(q_0, 1, Z_0) = (q_0, 1Z_0)$
3. $\delta(q_0, 0, 0) = (q_0, 00)$
4. $\delta(q_0, 1, 1) = (q_0, 11)$

This operation will store 0's or 1's in the stack.

Pop Operations:

1. $\delta(q_0, 0, 1) = (q_0, \wedge)$
2. $\delta(q_0, 1, 0) = (q_0, \wedge)$
3. $\delta(q_0, \wedge, Z_0) = (q_f, \wedge)$ (Accepting by null store)

This operation will remove 0's corresponding to 1's on input tape and vice-versa.

This is the required PDA.

Now the processing of the given string "011001" as follows:

$\delta(q_0, 011001, Z_0)$ |---- $(q_0, 11001, 0Z_0)$ |---- $(q_0, 1001, Z_0)$ |----- $(q_0, 001, 1Z_0)$ |----

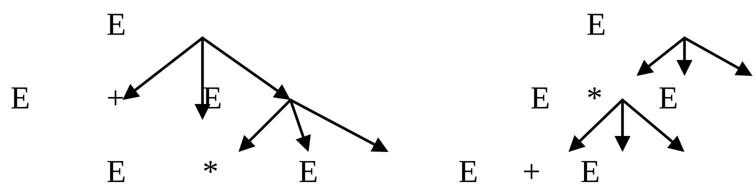
$(q_0, 01, Z_0) \vdash\!\text{----} (q_0, 1, 0Z_0) \vdash\!\text{-----} (q_0, \wedge, Z_0) = q_f$ (the final state).
Hence the given sting is accepted by the designed PDA.

b. What is ambiguity? Show that $S \to aS \mid Sa \mid a$ is an ambiguous grammar.

Answer:

Ambiguity in CFG: A grammar is said to be ambiguous if for any string we have two left or right most derivation trees.

For example, $E \to E + E \mid E * E$ can generate string $E + E * E$ in two ways



Two derivation trees with same yield

Now to show that $S \to aS \mid Sa \mid a$ is ambiguous we need to construct two different trees for same string say w = aaaa.

$S \to aS \to aaS \to aaaS \to aaaa$ (by $S \to aS \mid a$) …..(1)

$S \to Sa \to Saa \to Saaa \to aaaa$ (by $S \to Sa \mid a$) …..(2)

Hence there exist two different ie; one left most and another right most derivation trees for the given grammar, hence it is ambiguous.

Q.7 a. What are applications of pumping lemma in Chomsky's normal form?
Convert the given grammar into Chomsky's Nf.
$S \to ASB,\ A \to aAS \mid a,\ B \to SbS \mid bB$

Answer: Page Number 127 of Text Book

b. Find a reduced grammar equivalent to $G = (V_N, \Sigma, P, S)$ where set P is given as follows:

$$S \to AB,\ A \to a,\ B \to b \mid C,\ D \to c$$

Answer:

Step 1(Removal of extra variables)

Construction of V_N':

Let us construct $W_1 = \{A, B, D \mid$ as $A \to a$, $B \to b$ and $D \to c$ are productions with a terminal string on the R.H.S. $\}$

$W_2 = W_1 \cup \{X \in V_N \mid X \to \alpha$ for some $\alpha \in (W_1^* \cup \Sigma)\}$
$= \{S, A, B, D\}$

Similarly $W_3 = W_2 \cup \{X \in V_N \mid X \to \alpha$ for some $\alpha \in (W_2^* \cup \Sigma)\} = W_2 \cup \phi = W_2$

Therefore $V_N' = \{S, A, B, D\}$ and hence $P' = S \to AB,\ A \to a,\ B \to b,\ D \to c$.

Thus $G' = (V_N{}', \Sigma, P', S)$

Step 2(Removal of useless productions)

Construction of $V_N{}''$:

$W_1 = \{S\}$

$W_2 = W_1 \cup \{X \in V_N \cup \Sigma \mid$ there is a production $A \rightarrow \alpha$ with $A \in W_1$ and α containing $X\}$

$W_2 = \{S\} \cup \{A, B\} = \{S, A, B\}$

$W_3 = W_2 \cup \{a, b\} = \{S, A, B, a, b\}$

$W_4 = W_3$

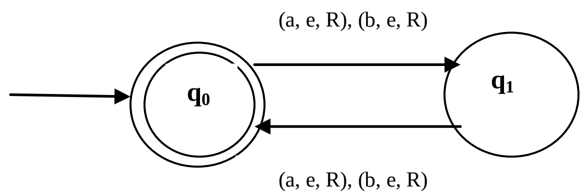
Thus $V_N{}'' = \{S, A, B\}$ and $P'' = S \rightarrow AB, A \rightarrow a, B \rightarrow b$.

Hence reduced grammar $G'' = \{V_N'' = \{S, A, B\}, \Sigma = (a, b), P'', S\}$

Q.8 a. Design a Turing machine that recognizes all strings of even length over $\Sigma = (a, b)^*$

Answer:

We construct the machine as follows:



To accept all strings over (a, b) of even length, we need two states to construct a loop from one to another. Let q_0 be the initial and final state. On reading either a or b at state q_0 machine goes to q_1 writes e (empty) on the input tape and moves to right direction. Similarly On reading either a or b at q_1 machine writes e (empty) on the input tape goes to state q_0 and moves to right direction. Hence it accepts all strings of even length over $(a, b)^*$.

Hence TM $M = \{(q_0, q_1), (a, b), q_0, \delta, \Gamma = (a, b, e), (L / R), \{q_0\}\}$ where the transition function δ is defined as above. It can be defined as transition table as follows:

State	Input	
	a	b
→ q_0	e,R,q_1	e,R,q_1
q_1	e,R,q_0	e,R,q_0

b. Write short note on universal Turing machine.

Answer:

A *universal* Turing machine is one that can be used to simulate any other Turing machine.

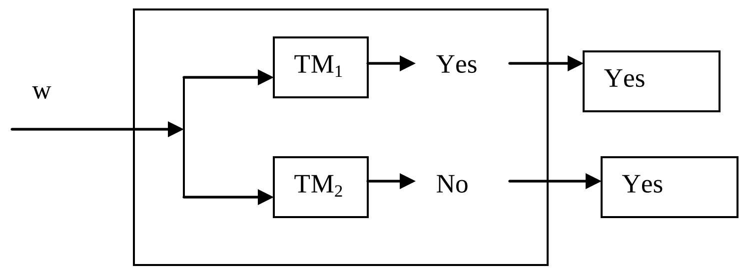
There exists a \universal" Turing machine U that, on input <M, w> where M is a TM and w is a sting over $(0, 1)^*$, simulates the computation of M on input w. Specifically:

1. U accepts <M, w> iff M accepts w
2. U rejects <M, w> iff M rejects w

Q.9 a. Prove that if a language L and it's complement L' are both recursively enumerable, then L is recursive.

Answer:

Let TM_1 and TM_2 accept L and L' respectively. Let us construct a turing machine TM which simulate TM_1 and TM_2 simultaneously. TM accepts w if TM_1 accepts it and rejects w if TM_2 will accepts it. Thus TM will always say either "yes" or "no", but not both. Note that there is not a priority limit on how long it may take before TM_1 or TM_2 accepts, but it is certain that one or the other will do so. Since TM is algorithm that accepts L, it follows that L is recursive.



\

b. Define Post corresponding Problem (PCP). Check whether the following instance has no solution over $\Sigma = \{0, 1\}$. X and Y be the lists of the three strings as follows:

	List A	List B
i	w_i	x_i
1	1	111
2	10111	10
3	10	0

Answer:

PCP: An instance of PCP consists of two lines of strings over some alphabet Σ; the two lists must be equal length. We generally refer to the A and B lists, and write $A = w_1, w_2, \ldots, w_k$ and $B = x_1, x_2, \ldots, x_k$, for some integer k. For each i, the pair (w_i, x_i) is said to be a corresponding pair.

We say this instance of PCP has a solution, if there is a sequence of one or more integers $i_1, i_2, \ldots, i_m$ that, when interpreted as indexes for strings in the A and B lists, yield the same string. That is, $w_{i1} w_{i2} \ldots w_{im} = x_{i1} x_{i2} \ldots x_{im}$. We say the sequence $i_1, i_2, \ldots, i_m$ is a solution to this instance of PCP, if so. The Post correspondence problem is:

"Given an instance of PCP, tell whether this has a solution."
Consider the problem given above

	List A	List B
i	w_i	x_i
1	1	111
2	10111	10
3	10	0

Let m = 4, $i_1 = 2$, $i_2 = 1$, $i_3 = 1$, and $i_4 = 3$, i.e; the solution is the list 2, 1, 1, 3. This list is a solution by concatenating the corresponding strings in order for the two lists. That is, $w_2w_1w_1w_3 = x_2x_1x_1x_3 = 101111110$.
Thus, in this case PCP has a solution.

Text Book

Introduction to Automata Theory, Languages and Computation, John E Hopcroft, Rajeev Motwani, Jeffery D. Ullman, Pearson Education, Third Edition, 2006