Chapter 1

Finite Automata and Regular Languages

LEARNING OBJECTIVES

- ☞ *Fundamentals*
- ☞ *Languages*
- ☞ *Operations*
- ☞ *Finite state machine*
- ☞ *NFA with ε-moves*
- ☞ *Conversion of NFA to DFA*
- ☞ *Minimization of DFA*
- ☞ *Equivalence between NFA and DFA*
- ☞ *Mealy and Moore machines*
- ☞ *Equivalence of Moore and Mealy machine*
- ☞ *Regular languages*
- ☞ *Constructing FA for given RE*
- ☞ *Pumping lemma for regular sets*
- ☞ *Closure properties of regular sets*
- ☞ *Regular grammar*

Fundamentals

Alphabet: An alphabet is a finite non-empty set of symbols.

Example: Portion of a calculator: $\{0, 1, 2, 3 \ldots 9, \div, =, -, +, \times, (,)\}$

Note: 1. At least one symbol is necessary.

2. 'Σ' denote Alphabet.

String: A string over an alphabet 'A' is a finite ordered sequence of symbols from 'A'. The length of string is number of symbols in string, with repetitions counted.

Example: If $\Sigma = \{0 - 9, \div, =, -, +, \times (,)\}$ then Strings valid: $12 + 34$, 90×10, $(1 + 2) \times (1 \div 3)$

Strings Invalid: sin (45), log (10) etc. These strings are not valid because sin ( ), log ( ) are not defined over the alphabet set.

Note: Repetitions are allowed.

Length of $|12 + 34| = 5(1, 2, +, 3, 4)$

- The Empty string denoted by 'ε', is the (unique) string of length zero.

Note: Empty string, $\varepsilon \neq$ empty set, $\varnothing$.

- If S and T are sets of strings, then $ST = \{xy | x \in S \text{ and } y \in T\}$

Given an alphabet A,

$A^o = \{\varepsilon\}$

$A^{n+1} = A.A^n$

$\ldots$

$$A^* = \bigcup_{n=0}^{\infty} A^n$$

Languages

- A language 'L' over Σ is any finite or infinite set of strings over Σ.
- The elements in L are strings – finite sequences of symbols.
- A language which does not contain any elements is called 'empty language'.

Note: Empty language, $\{\ \} \neq \{\varepsilon\}$, empty string because $\{\ \} = \varnothing \neq \varepsilon$ i.e., Empty language resembles empty set i.e., $\varnothing$.

- A language L over an alphabet A is subset of A^* i.e., $L \subset A^*$.

Example 1: Language (L) for strings that consists of only 0's or only 1's and have an odd length over alphabet {0, 1} is

(A) {0, 1, 00, 11, 000, 111 …}
(B) {00, 11, 01, 10 …}
(C) {000, 101, 110, 111 …}
(D) {0, 1, 000, 111, 11111, 00000 …}

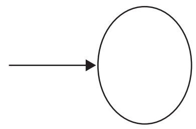
Solution: (D)

Only 0's → should have only 0's. It should not be combination of 0's and 1's.

Only 1's → should have only 1's. It should not be combination of 0's and 1's.

Odd length → only odd number of 0's or odd number of 1's i.e., length of string should be odd.

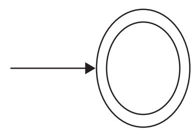
An Empty Languages An empty language is a language which does not accept any strings includinge. The Finite automata for empty language can be represented as



(i.e., One state, non-accepting and no transitions).

A language which only accepts (ε)

E: The language which only accepts 'ε' can be represented as



This machine accepts E – only.

Σ^***:** The set of all strings over an alphabet Σ will be denoted by Σ^*.

Σ^+: This will denote the set $\Sigma^* - \{\varepsilon\}$.

Ex: If $\Sigma = \{0, 1\}$ then

$\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, \ldots\}$

$\Sigma^+ = \{0, 1, 00, 01, 10, 11, 000, 001,\}$

Operations

Operations on strings

1. **Concatenation:** Combines two strings by putting one after other.

Example 2: Two strings are defined as x = java, y = script. The concatenation ($x.y$) of two strings results in _______.

(A) scriptjava (B) javascript
(C) jascriptva (D) scrijavapt

Solution: (B)

$x.y$ = java.script = javascript

Note: Concatenation of empty string with any other string gives string itself.

i.e., $x.\varepsilon = \varepsilon.x = x$

2. **Substring:** If 'w' is a string, then 'v' is a substring of 'w' if there exists string x and y such that $w = xvy$.

'x' is called 'prefix' and y is called suffix of w.

Example 3: String, w = 'gymnastics' is defined with prefix, x = '*gym*' and suffix, y = '*cs*'. The substring of the given string is _______

(A) nasti (B) mnas
(C) gymnastics (D) ics

Solution: (A)

Because, $w = xvy$

⇒ gymnastics = gymvcs

∴ v = nasti

3. **Kleen star operation:** Let 'w' be a string, w^* is set of strings obtained by applying any number of concatenations of w with itself, including empty string.

Example: $a^* = \{\varepsilon, a, aa, aaa, \ldots\}$

4. **Reversal:** If 'w' is a string, then w^R is reversal of string spelled backwards.

Rules:

- $x = (x^R)^R$
- $(xz)^R = z^R \cdot x^R$

Example 4: A string, x is defined as, x = butter. Then $(x^R)^R$ is _______

(A) butter (B) rettub
(C) butret (D) retbut

Solution: (A)

$x \rightarrow$ butter

$x^R \rightarrow$ rettub

$(x^R)^R \rightarrow$ butter.

Operations on languages

1. **Union:** Given some alphabet Σ, for any two languages, L_1, L_2 over Σ, the union $L_1 \cup L_2$ of L_1 and L_2 is the language, $L_1 \cup L_2 = \{w \in \Sigma^* | w \in L_1 \text{ or } w \in L_2\}$
2. **Intersection:** Given some alphabet Σ, for any two languages L_1, L_2 over Σ, the intersection $L_1 \cap L_2$ of L_1 and L_2 is language, $L_1 \cap L_1 = \{w \in \Sigma^* | w \in L_1 \text{ and } w \in L_2\}$
3. **Difference:** Given some alphabet Σ, for any two languages L_1, L_2 over Σ, the difference $L_1 - L_2$ of L_1 and L_2 is language, $L_1 - L_2 = \{w \in \Sigma^* | w \in L_1 \text{ and } w \notin L_2\}$

Note: Difference is also called 'Relative Complement.'

A special case of difference is obtained when $L_1 = \Sigma^*$, in which case. Complement $\bar{L}$ of language, L is defined as, $\bar{L} = \{w \in \Sigma^* | \, w \notin L\}$

4. **Concatenation:** Given an alphabet Σ, for any two languages L_1, L_2 over Σ, the concatenation L_1 L_2 of L_1 and L_2 is language

$L_1L_2 = \{w \in \Sigma^* | \exists\, u \in L_1, \exists\, v \in L_2, w = uv\}$

Properties:

$L\varnothing = \varnothing = \varnothing L$

$L\,\{\varepsilon\} = L = \{\varepsilon\}\,$L

$(L_1 \cup \{\varepsilon\})L_2 = L_1L_2 \cup L_2$

$L_1\,(L_2 \cup \{\varepsilon\}) = L_1L_2 \cup L_1$

$L^n\,L = LL^n = L^{n+1}$

Note: $L_1L_2 \neq L_2L_1$

Example 5: Let $L_1 = \{00, 11\}$, $L_2 = \{01, 10\}$. Then $L_1 o L_2$ = _______

(A) {00, 11, 01, 10}
(B) {0001, 0010, 1101, 1110}
(C) {0001, 0010, 11, 01, 10}
(D) {00, 1101, 1110, 11, 10}

Solution: (B)

$L_1 o L_2$ = {00, 11} o {01, 10} = {00.01, 00.10, 11.01, 11.10} = {0001, 0010, 1101, 1110}

5. **Kleen * closure (L*):** Given an alphabet Σ, for any language L over Σ, the * closure L^* of L is language, $L^* = U_{n \geq 0} L^n$

6. **Kleen + closure (L^+):** The kleen +closure, L^+ of L is the language, $L^+ = U_{n\geq 1} L^n$

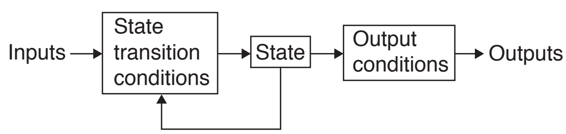
$L^* = L^0 \cup L^1 \cup L^2 \cup \ldots L^n \cup \ldots$
$L^+ = L^1 \cup L^2 \cup L^3 \ldots \cup L^n \cup \ldots$

Properties:

$\varnothing^* = \{\varepsilon\}$
$L^+ = L^*L$
$(L^*)^* = L^*$
$L^* L^* = L^*$

Finite State Machine (FSM)

- FSM is simplest computational model of limited memory computers.
- FSM is designed to solve decision problems i.e., to decide whether given input satisfies certain conditions.
- The next state and output of a FSM is a function of input and of current state.

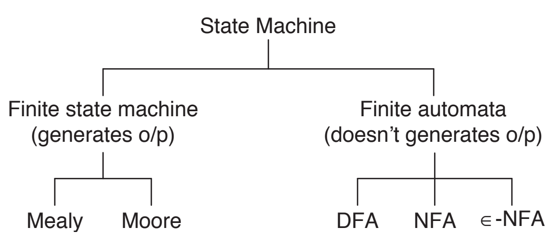


Types of FSM:

1. Melay machine.
2. Moore machine

Finite Automata (FA):

- FA is a state machine that comprehensively captures all possible states and transitions that a machine can take while responding to a stream (sequence) of input symbols.
- FA is recognizer of 'regular languages'.



Types of FA

1. Deterministic Finite Automata (DFA):

- DFA machine can exists in only one state at any given time.
- DFA is defined by 5-tuple: $\{Q, \Sigma, q_0, F, \delta\}$, where

$Q \to$ Finite number of states (elements)
$\Sigma \to$ Finite set of symbols (alphabets)
$q_o \to$ Start/Initial state
$F \to$ Set of final states.
$\delta \to$ Transition function, which is a mapping between

$$\delta: Q \times \Sigma \to Q.$$

How to use DFA:

Input: A word w in Σ^*
Question: Is w acceptable by DFA?

Steps:

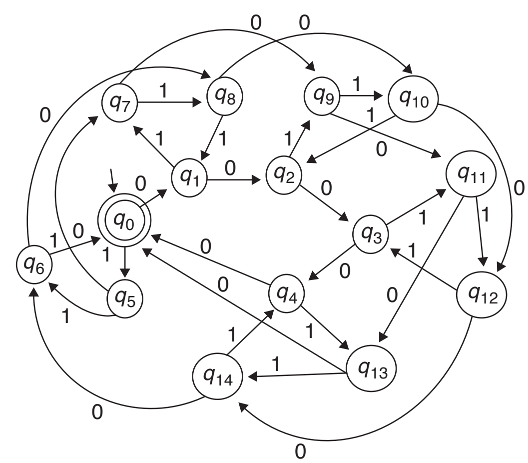
- Start at 'initial state', q_o.
- For every input symbol in sequence w, do.
- Compute the next state from current state, given the current input symbol in w and transition function.
- If after all symbols in 'w' are consumed, the current state is one of the final states (f) then accept 'w';
- Otherwise, reject w.

Transition diagram: State machines are represented by directed graphs called transition (state) diagrams.

- The vertices denoted by single circle represent the state and arcs labeled with input symbol correspond to transition.
- The final states are represented with double circles.

Transition Table: Transition function can be represented by tables.

Example 6: The following finite state machine accepts all those binary strings in which the numbers of 0's and 1's are respectively.



(A) Divisible by 3 and 2 (B) Odd and even
(C) Divisible by 5 and 3 (D) Divisible by 2 and 3

Solution: (C)
Number of 0's is divisible by 5.
Number of 1's is divisible by 3.

Table *Transition Table*

Current State	0	1
$\to q_0$	q_1	q_5
q_1	q_2	q_7
q_2	q_3	q_9
q_3	q_4	q_{11}
q_4	q_0	q_{13}
q_5	q_7	q_6
q_6	q_8	q_0
q_7	q_9	q_8
q_8	q_{10}	q_1

q_9	q_{11}	q_{10}
q_{10}	q_{12}	q_2
q_{11}	q_{13}	q_{12}
q_{12}	q_{14}	q_3
q_{13}	q_0	q_{14}
q_{14}	q_6	q_4

Note: Minimum number of states for k-divisibility is k-states.
In above example, $q_0 - q_{14} \rightarrow 15$ – states.
$\therefore \quad 5 \times 3 = 15$
The given binary strings have number of 0's divisible by 5 and number of 1's divisible by 3.

2. Non-deterministic finite Automata (NFA):

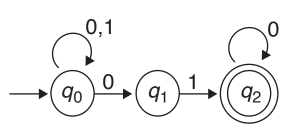
- The machine can exist in multiple states at the same time.
- Each transition function maps to a set of states.
- NFA is defined by 5-tuple: $\{Q, \Sigma, q_0, F, \delta\}$, where

$Q \rightarrow$ Finite number of states (elements)
$\Sigma \rightarrow$ Finite set of symbols. (Alphabets)
$q_o \rightarrow$ Start/Initial state
$F \rightarrow$ Set of final states.
$\delta \rightarrow$ Transition function which is a mapping between
$\delta = Q \times \Sigma \rightarrow 2^Q$

How to use NFA:
Input: a word w in Σ^*
Question: Is w accepted by NFA?
Steps:

- Start at 'start state' q_0.
- For every input symbol in the sequence, w does.
- Determine all possible next states from current state, given the current input symbol in w and transition function.
- If after all symbols in w are consumed, at least one of the current states is a final state then accept w.
- Otherwise, reject w.

Example 7: What is the language, L generated by the below NFA, given strings defined over alphabet, $\Sigma = \{0, 1\}$.



(A) Strings that end with '0'
(B) Strings that start with '0' and end with '0'
(C) Strings that contain '01' as substring
(D) Strings that contain '01' as substring and end with '0'

Solution: (D)

State	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\varnothing$	q_2
q_2	$\{q_2\}$	$\varnothing$

String: 0100100

$$q_0 \xrightarrow{0} \{q_0, q_1\}$$
$$q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_0 \xrightarrow{0} \{q_0, q_1\}$$
$$q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_0 \xrightarrow{0} q_0 \xrightarrow{0} \{q_0, q_1\}$$
$$\xrightarrow{q_0 1} q_0 \xrightarrow{0} q_0 \xrightarrow{0} \{q_0, q_1\} \text{ (Non-deterministic)}$$
$$q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_0 \xrightarrow{0} q_0 \xrightarrow{0} \{q_0, q_1\}$$
$$q_1 \xrightarrow{1} q_2 \xrightarrow{0} q_2 \xrightarrow{0} q_2$$

Table 2 *Difference between NFA and DFA*

DFA	NFA
1. All transitions are deterministic i.e., each transition leads to exactly one state.	1. Transitions could be non-deterministic i.e., a transition could lead to a subset of states.
2. For each state, the transition on all possible symbols should be defined.	2. For each state, not all symbols necessarily have to be defined.
3. Accepts input if last state is in 'F'.	3. Accepts input if one of last states is in 'F'.
4. Practical implementation is feasible.	4. Practical implementation has to be deterministic (so needs conversion to DFA).

Relation between DFA and NFA

- A language 'L' is accepted by a DFA if and only if it is accepted by a NFA.
- Every DFA is special case of a NFA.

Example 8: Let N_f and D_f denote the classes of languages accepted by non-deterministic finite automata and deterministic finite automata respectively. Which one of following is true?

(A) $D_f \subset N_f$ (B) $D_f \supset N_f$
(C) $D_f = N_f$ (D) $D_f \in N_f$

Solution: (C)
According to 'subset construction', every language accepted by NFA is also accepted by some DFA.
$\therefore \quad D_f = N_f$

NFA WITH $\in$-MOVES

- $\in$-transitions in finite automata allows a state to jump to another state without consuming any input symbol.

Conversion and Equivalence:

$\in$-NFA $\rightarrow$ NFA $\rightarrow$ DFA

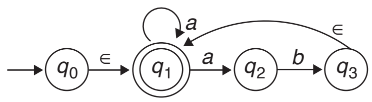
NFA without $\in$-moves:

- Two FA, $N_\in$ and N are said to be equivalent, if $L(N_\in) = L(N)$ i.e., any language described by some $N_\in$, there is an N that accepts the same language.
- For $N_\in = (Q, Z, \delta, q_0, F)$ and $N = (Q, \Sigma', \delta', q_0, F')$, Find
- $\delta'(q, a) = \in$-closure $(\delta(\in\text{-closure}(q), a))$

- $F' = \{F \cup \{q_0\}\}$, if $\in$-closure (q_0) contains a member of $F = F$, otherwise.

Note: When transforming $N_\in$ to N, only transitions are required to be changed and states remains same.

Example 9: Consider following NFA with $\in$-moves.



If given NFA is converted to NFA without $\in$-moves, which of following denotes set of final states?

(A) $\{q_0, q_1\}$ (B) $\{q_1, q_2\}$
(C) $\{q_1, q_2, q_3\}$ (D) $\{q_1\}$

Solution: Let $N = (Q, \Sigma^1, \delta^1, q_0, F^1)$
$F^1 = F \cup \{q_0\}$
ε-closure $(q_0) = \{q_0, q_1\}$
$\therefore F^1 = \{q_1\} \cup \{q_0, q_1\} = \{q_0, q_1\}$
Conversion $N_\in \to N$:
To compute, δ^1
$\in$-closure $(q_0) = \{q_0, q_1\}$, $\in$-closure $(q_3) = \{q_3, q_1\}$
$\delta^1(q_0, a) = \{q_1, q_2\}$, $\delta^1(q_0, b) = \varnothing$, $\delta^1(q_2, a) = \varnothing$.
$\delta^1(q_1, a) = \{q_1, q_2\}$, $\delta^1(q_1, b) = \varnothing$, $\delta^1(q_2, b) = \{q_1, q_3\}$
$\delta^1(q_3, a) = \{q_1, q_2\}$, $\delta^1(q_3, b) = \varnothing$

Table 3 *Transition Table*

State \ Input	a	b
$\to q_0$	$\{q_1, q_2\}$	$\varnothing$
q_1	$\{q_1, q_2\}$	$\varnothing$
q_2	$\varnothing$	$\{q_1, q_3\}$
q_3	$\{q_1, q_2\}$	$\varnothing$

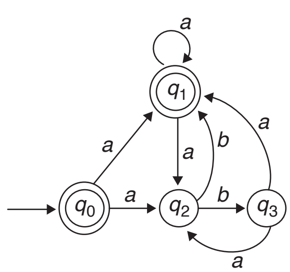


Figure 1 Transition diagram

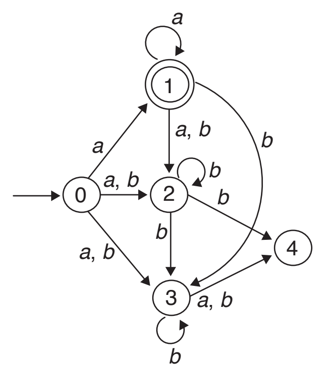
Conversion of NFA to DFA

Let a NFA be defined as, $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$
The equivalent DFA, $D = (Q_D, \Sigma, \delta_D, q_0, F_D)$ where:
Step I: $Q_D = 2^{Q_N}$; i.e., Q_D is set of all subsets of Q_N i.e., it is power set of Q_N.
Step II: F_D is set of subsets S of Q_N such that $S \cap F_N \neq \varnothing$. i.e., F_D is all sets of N's states that include atleast one accepting state of N.
Step III: For each set, $S \leq Q_N$ and for each input symbol a in Σ: $\delta_D(S, a) = \cup_{P \in S} \delta_N(P, a)$

That is, to compute $\delta_D(S, a)$, look at all states P in S, see what states N goes to starting from P on input a, and take the union of all those states.

Note: For any NFA, N with 'n' states, the corresponding DFA, D can have 2^n states.

Example 10: What is the number of final states in DFA constructed from the given NFA?



(A) 1 (B) 2
(C) 3 (D) 4

Solution:

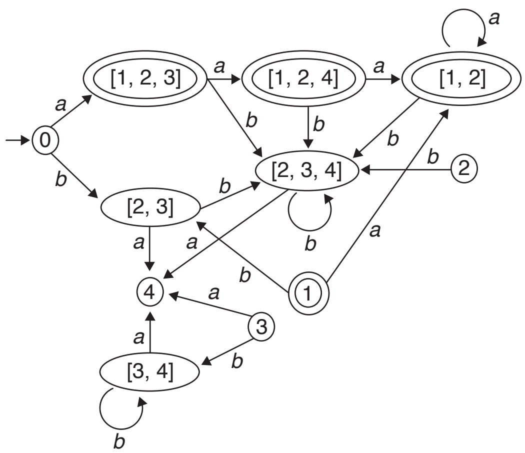
Table 4 *Transition Table of NFA*

State \ Input	a	b
$\to 0$	{1, 2, 3}	{2, 3}
1	{1, 2}	{2, 3}
2	$\varnothing$	{2, 3, 4}
3	{4}	{3, 4}
4	$\varnothing$	$\varnothing$

Table 5 *Transition Table of DFA*

State \ Input	a	b
$\to 0$	[1, 2, 3]	[2, 3]
1	[1, 2]	[2, 3]
2	$\varnothing$	[2, 3, 4]
3	4	[3, 4]
4	$\varnothing$	$\varnothing$
[1, 2]	[1, 2]	[2, 3, 4]
[2, 3]	[4]	[2, 3, 4]
[3, 4]	[4]	[3, 4]
[1, 2, 3]	[1, 2, 4]	[2, 3, 4]
[1, 2, 4]	[1, 2]	[2, 3, 4]
[2, 3, 4]	[4]	[2, 3, 4]

Hence final states in obtained DFA is '4'.
DFA is: Choice (D)

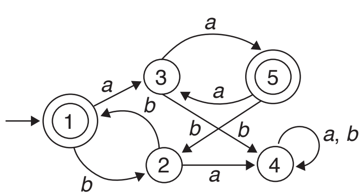


Minimization of DFA

Given a DFA, $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA, $M' = (Q', \Sigma', \delta', q_0', F')$ as follows

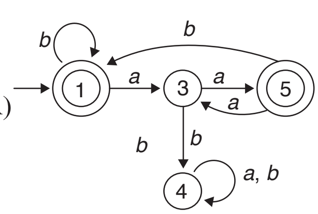
1. Remove all inaccessible states. All states that are unreachable from the initial state are removed.
2. Consider all pairs of states (p, q), If $p \in F$ and $q \in F$ or vice versa mark the pair (p, q) as distinguishable.
3. Repeat until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = p_a$ and $\delta(p, q) = q_a$. If the pair (p_a, q_a) is marked as distinguishable mark (p, q) as distinguishable.
4. Find the sets of all indistinguishable states, say $\{q_i, q_j, \cdots q_k\}$, $\{q_\ell, q_m, \cdots q_n\}$, etc. For each set $\{q_i, q_j, \ldots q_k\}$ of such indistinguishable states, create a state labelled $ij \ldots k$ for M.
5. For each transition rule of M of the from $\delta(q_r, q) = q_p$, find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots q_k\}$ and $q_p \varepsilon \{q_\ell, q_m, \cdots q_n\}$, add a rule to δ: $\delta'(ij \ldots k, a) = \ell m \ldots n$.

Example 11: A DFA with alphabet $\Sigma = \{a, b\}$ is given below:

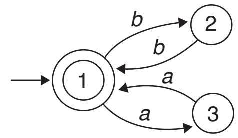


Which of the following is valid minimal DFA which accepts same language as given DFA?

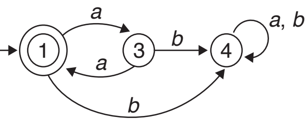
(A)



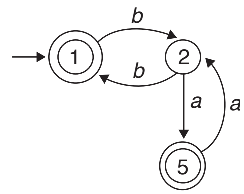
(B)



(C)



(D)



Solution: (B)

Initially, $\{1, 5\}, \{2, 3, 4\}$

Depending on next states and inputs, the partitions of states can be as: $\{\{1, 5\}, \{2\}, \{3\}, \text{and } \{4\}\}$

Since, 1 to 5 have same transition, unite $\{1, 5\}$

State 4 is dead state $\rightarrow$ It has transition only to itself. Since, $\{2\}, \{3\}$ are singletons, they exist.

$\therefore$ States in minimized DFA are $\{1, 2, \text{and } 3\}$

$\{1\} \rightarrow \{1, 5\}$

For transitions, since $1 \xrightarrow{a} 3$, $1 \xrightarrow{b} 2$ in given DFA, in minimized DFA, transitions are added from $1 \xrightarrow{a} 3$, $1 \xrightarrow{b} 2$. Also, since $2 \xrightarrow{b} 1$, $3 \xrightarrow{a} 1$ in given DFA, the minimized DFA, transitions are added from $2 \xrightarrow{b} 1$, $3 \xrightarrow{a} 1$.

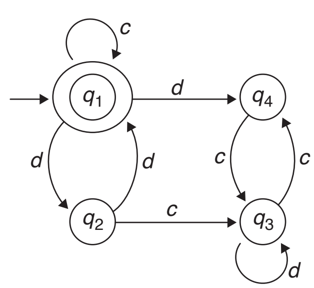
Equivalence Between NFA and DFA

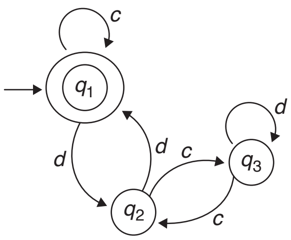
There is a DFA_D for any NFA_N i.e., $L(D) = L(N)$.

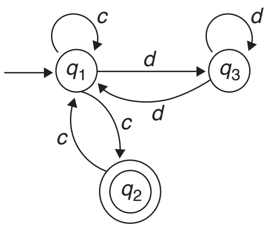
Construction:

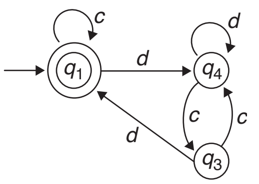
- In DFA or NFA, whenever an arrow is followed, there is a set of possible states. This set of states is a subset of Q.
- Track the information about subsets of states that can be reached from initial state after following arrows.
- Consider each subset of states of NFA as a state of DFA and every subset of states containing a final state as a final state of DFA.

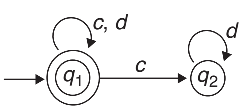
Example 12: Which of following is equivalent DFA for the NFA given below:



(A) 

(B) 

(C) 

(D) 

Solution: (A)

Table 6 *Transition Table of NFA*

δ	**c**	**d**
$\rightarrow q_1$	q_1	$\{q_2, q_4\}$
q_2	q_3	q_1
q_3	q_4	q_3
q_4	q_3	$\varnothing$

Table 7 *Transition Table of DFA*

δ	**c**	**d**
$\rightarrow q_1$	q_1	q_2
q_2	q_3	q_1
q_3	q_2	q_1

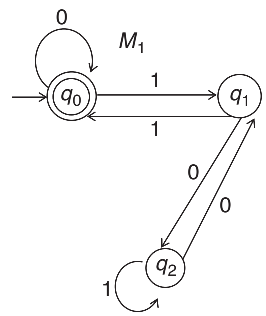
Table 8 *Common Table*

δ	**c**	**d**
(q_1, q_1)	(q_1, q_1)	(q_2, q_4, q_2)
(q_2, q_2)	(q_3, q_3)	(q_1, q_1)
(q_3, q_3)	(q_4, q_2)	(q_3, q_3)
(q_4)	q_3	$\varnothing$

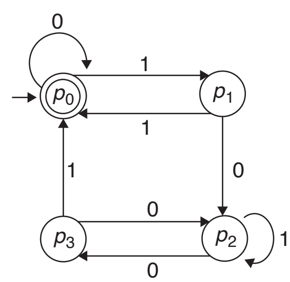
Equivalence of Finite Automatas:

- Two automatas A and B are said to be equivalent if both accept exactly the same set of input strings.
- If two automatas M_1 and M_2 are equivalent then
  (i) If there is a path from the start state of M_1 to a final state of M_1 labeled $a_1a_2 \ldots a_k$ then there is a path from the start state of M_2 to the final state of M_2 labeled $a_1a_2 \ldots a_k$.
  (ii) If there is a path from the start state of M_2 to a final state M_2 labeled $b_1b_2 \ldots b_i$ then there is a path from the start state of M_1 to the final state of M_1 labeled $b_1b_2 \ldots b_i$.

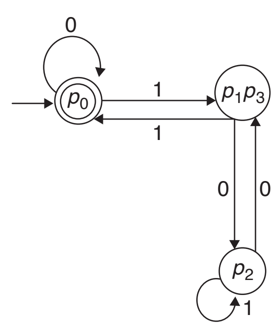
Example:



M_2:



In M_2, states p_1 and p_3 are equivalent (as both are reaching either final or non-final states with same input). After minimizing M_2, we will get



$\therefore$ M_1 and M_2 are equivalent.

Union: The union of two languages L and M is the set of strings that are in both L and M.
Ex: $L = \{0, 1\}$, $M = \{111\}$
$L \ U \ M = \{0, 1, 111\}$.

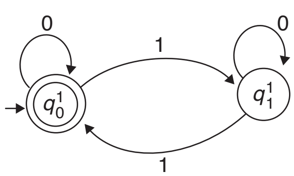
Concatenation: The concatenation of Languages L and M is the set of strings that can be formed by taking any string in L and concatenating it with any string in M.

Example: $L = \{0, 1\}$, $M = \{\varepsilon, 010\}$
$LM = \{0, 1, 0010, 1010\}$.

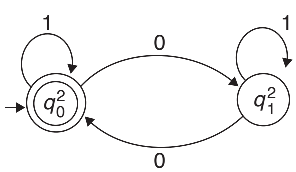
Closure, Star or Kleen star of a language L:
Kleen star is denoted as L^*. It represents the set of strings that can be formed by taking any number of strings from L with repetition and concatenating them. It is a Unary operator. L^0 is the set; we can make selecting zero strings from L.
$L^0 = \{\varepsilon\}$
L^1 is the language consisting of selecting one string from L.
L^2 is the language consisting of concatenations selecting two strings from L.
…
L^* is the union of $L^0, L^1, \ldots L^\infty$.
Ex: $L = \{0,10\}$
$L^* = \{0,00,000,10,010, \ldots\}$

Intersection:
Let two DFAs M_1 and M_2 accept the languages L_1 and L_2.
$M_1 = (Q_1, \Sigma, \delta_1, q_0^1, .F_1)$
$M_2 = (Q_2, \Sigma, \delta_{2,} q_0^2, F_2)$
The intersection of M_1 and M_2 can be given as
$M = (Q_1, \Sigma, \delta, q_0\ F)$
Q = Pairs of states, one from M_1 and one from M_2 i.e.,
$Q = \{(q_1, q_2) \mid q_1 \in Q_1 \text{ and } q_2 \in Q_2\}$
$Q = Q_1 \times Q_2$.
$q_0 = (q_0^1, q_0^2)$
$\delta\,(q_i^1\ q_j^2), x) = (\delta_1(q_1^1, x), \delta_2(q_j^2, x))$
$F = \{(q_1, q_2) \mid q_1, \in F_1 \text{ and } q_2 \in F_2\}$

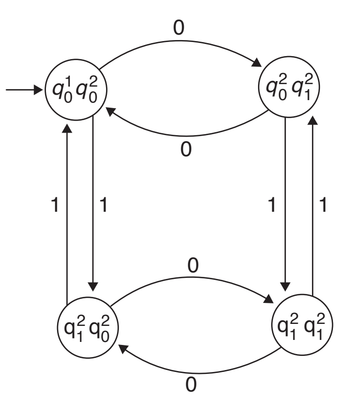
Example:
M_1: Strings with even number of 1's.



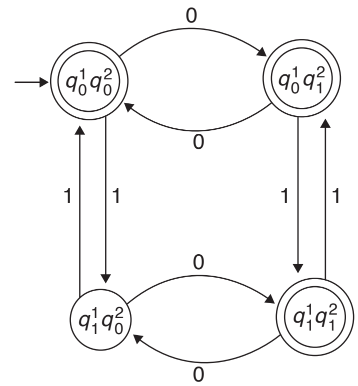
M_2: Strings with odd number of 0's.



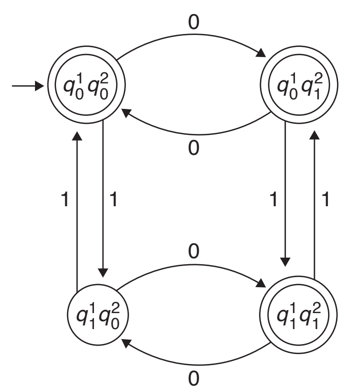
$M_1 \cap M_2$: Strings with even number of 1's and odd number of 0's.



Union of M_1 and M_2:



Difference: The difference of L_1 and L_2 can be given as $L_1 - L_2$ with $M = (Q, \Sigma, \delta, q_0, F)$.
$Q = Q_1 \times Q_2$
$q_0 = (q_0{}^1, q_0{}^2)$
$\delta((q_i{}^1, q_j{}^2), x) = (\delta_1\,(q_i{}^1, x), \delta_2(q_j{}^2, x))$
$F = \{(q_1, q_2) \mid q_1, \in F_1 \text{ and } q_2 \notin F_2\}$



Reversing a DFA:

- M is a DFA which recognizes the language L.
- M^R will accept the language L^R.

To construct M^R:

- Reverse all transitions
- Turn the start state to final state
- Turn the final states to start state.
- Merge states and modify the FA,

such that the resultant contain a single start state.

Mealy and Moore Machines

Moore Machine

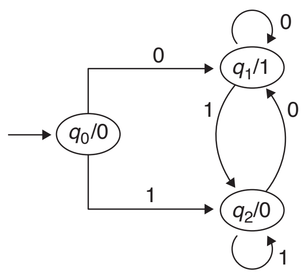
A moore machine is a finite state machine, where outputs are determined by current state alone.

A Moore machine associates an output symbol with each state and each time a state is entered, an output is obtained simultaneously. So, first output always occurs as soon as machine starts.

Moore machine is defined by 6-tuples:
$(Q, \Sigma, \delta, q_0, \Delta, \lambda)$, where
$Q \rightarrow$ Finite set of states
$\Sigma \rightarrow$ Finite set of input symbols
$\Delta \rightarrow$ It is an output alphabet
$\delta \rightarrow$ Transition function, $Q \times \Sigma \rightarrow Q$ (state function)
$\lambda \rightarrow$ Output function, $Q \rightarrow \Delta$ (machine function)
$q_0 \rightarrow$ Initial state of machine

Note: The output symbol at a given time depends only on present state of moore machine.

Example 13: The language generated by the following moore machine is:



(A) 2's complement of binary number.
(B) 1's complement of binary number.
(C) Has a substring 101.
(D) Has a substring 110.

Solution: (B)
Binary number: 1011
1's complement: 0100

$$q_0 \xrightarrow{1/0} q_2, \xrightarrow{0/1} q_1 \xrightarrow{1/0} q_2 \xrightarrow{1/0} q_2$$

$1 \rightarrow 0,\ 0 \rightarrow 1,\ 1 \rightarrow 0,\ 1 \rightarrow 0$

Mealy Machine

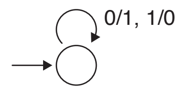
- A mealy machine is a FSM, where outputs are determined by current state and input.
- It associates an output symbol with each transition and the output depends on current input.
- Mealy machine is defined on 6-tuples: $(Q, \Sigma, \delta, q_0, \Delta, \lambda)$, where

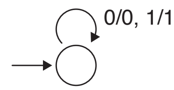
Q – Finite set of states.
Σ – Finite set of input symbols.
δ – $(Q \times \Sigma \rightarrow Q)$ is transition function.
$q_0 \rightarrow q_0 \in Q$ is initial state.
$\Delta \rightarrow$ Finite set of output symbols.
$\lambda \rightarrow$ Output function, $\lambda(Q \rightarrow \Delta)$

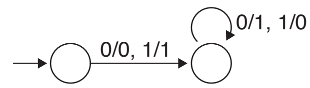
Note: In Moore machine, for input string of length n, the output sequence consists of $(n + 1)$ symbols.

In Mealy machine, for input string of length n, the output sequence also consists of 'n' symbols.

Example 14: Let $(\text{Me})^2$ mean that given a Mealy machine, an input string is processed and then output string is immediately fed into the machine (as input) and reprocessed. Only this second resultant output is considered as the final output of $(\text{Me})^2$. If final output string is same as original input string then $(\text{Me})^2$ has an identity property. Consider following machines.

(i) 

(ii) 

(iii) 

Which of above machines have identity property?
(A) (i) only
(B) (i) and (ii) but not (iii)
(C) (i) and (iii) but not (ii)
(D) All have identity property

Solution: (D)

(i) Consider i/p string

0 1 0 1
↓ ↓ ↓ ↓
o/p string 1 0 1 0 (Input string reprocessed)
↓ ↓ ↓ ↓
0 1 0 1 → o/p string

(ii)

i/p string: 0 1 1 0
↓ ↓ ↓ ↓
o/p string: 0 1 1 0
(i/p string) ↓ ↓ ↓ ↓
o/p string: 0 1 1 0

(iii)

i/p string: 1 0 0 1
↓ ↓ ↓ ↓
o/p string: 1 1 1 0
(i/p string) ↓ ↓ ↓ ↓
o/p string: 1 0 0 1

Equivalence of Moore and Mealy machine

(a) Mealy machine equivalent to Moore machine:
If $M_1 = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$ is a Moore machine, then there is a Mealy machine M_2 equivalent to M_1.

Proof: Let $M_2 = (Q, \Sigma, \Delta, \delta, \lambda^1, q_0)$ and define $\lambda^1(q, a)$ to be $\lambda(\delta(q, a))$ for all states q and input symbol 'a'.

Then M_1 and M_2 enter the same sequence of states on the same input, and with each transition M_2 emits the o/p that M_1 associates with the state entered.
Let us consider Mealy Machine

Present State	Next State			
	a = 0		a = 1	
	Input State	Output	Input State	Output
$\to q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

To convert the Mealy machine to Moore machine,

- We look into the next state column for any state, say q_i and determine the number of different outputs associated with q_i in next column.
- Split q_i into several different states, the number of such states being equal to the number of different outputs associated with q_i.

Present State	Next State			
	a = 0		a = 1	
	Input State	Output	Input State	Output
$\to q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}	1	q_1	1
q_{40}	q_{41}	1	q_3	0
q_{41}	q_{41}	1	q_3	0

- The pair of states and outputs in the next state column can be rearranged as:

Present state	Next State		output
	a = 0	a = 1	
$\to q_1$	q_3	q_{20}	1
q_{20}	q_1	q_{40}	0
q_{21}	q_1	q_{40}	1
q_3	q_{21}	q_1	0
q_{40}	q_{41}	q_3	0
q_{41}	q_{41}	q_3	1

Moore machine equivalent to Mealy machine

Let $M_1 = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$ be a Mealy machine. Then there is a machine M_2 equivalent to M_1

Proof: Let $M_2 = (QX\Delta, \Sigma, \Delta, \delta^1, \lambda^1, [q_0, b_0])$, where b_0 is an arbitrary selected member of Δ.

That is, the states of M_2 are pairs $[q, b]$ consisting of a state of M_1 and output symbol, Define $\delta^1([q, b], a) = [\delta(q, a), \lambda, (q, a)]$ and $\lambda^1([q, b]) = b$.

The second component of a state $[q, b]$ of M_2 is the output made by M_1 on some transition into state q.

Only the first components of M_2's states determine the moves made by M_2.

Every induction on 'n' shows that if M_1 enters states $q_0, q_1 \ldots q_n$ on inputs $a_1, a_2 \ldots a_n$ and emits output $b_1, b_2, \ldots b_n$ then M_2 enters states $[q_0, b_0], [q_1, b_1] \ldots [q_n, b_n]$ and emits outputs $b_0, b_1 \ldots b_n$.

Let us consider the Moore machine

Present State	Next State		Output
	a = 0	a = 1	
$\to q_0$	q_3	q_1	0
q_1	q_1	q_2	1
q_2	q_2	q_3	0
q_3	q_3	q_0	0

- To convert Moore into Mealy machine, we must follow the reverse procedure of converting Mealy machine into Moore machine.
- For every input symbol we form, the pair consisting of the next state and the corresponding output and reconstruct the table for Mealy machine.
- For example, the state q_3 and q_1 in the next state column should be associated with outputs 0 and 1, respectively.

The Transition table for Mealy machine is:

Present state	Next State			
	a = 0		a = 1	
	state	output	state	output
$\to q_0$	q_3	0	q_1	1
q_1	q_1	1	q_2	0
q_2	q_2	0	q_3	0
q_3	q_3	0	q_0	0

Regular Languages

The set of regular languages over an alphabet Σ is defined recursively as below. Any language belonging to this set is a regular language over Σ.

Definition of set of regular languages

- Basis clause: $\varnothing$, $\{\varepsilon\}$, $\{a\}$ for any symbol $a \in \Sigma$, are regular languages.
- Inductive clause: If L_r and L_s are regular languages, then $L_r \cup L_s$, $L_r \cdot L_s$, L_r^* are regular languages.
- External clause: Nothing is a regular language, unless it is obtained from above two clauses.

Regular language: Any language represented by regular expression(s) is called a regular language.

Ex: The regular expression a^* denotes a language which has $\{\varepsilon, a, aa, aaa, \ldots\}$

Regular expression

- Regular expressions are used to denote regular languages.
- The set of regular expressions over an alphabet Σ is defined recursively as below. Any element of that set is a regular expression.

- Basis clause: $\varnothing$, $\in$, a are regular expression corresponding to languages $\varnothing$, $\{\in\}$, $\{a\}$ respectively where a is an element of Σ.
- Inductive clause: If r and s are regular expression corresponding to languages L_r and L_s then $(r + s)$, (rs) and (r^*) are regular expressions corresponding to the languages $L_r \cup L_s$, $L_r \cdot L_s$ and L_r^* respectively.
- External clause: Nothing is a regular expression, unless it is obtained from above two clauses.

Closure property of regular expressions The iteration or closure of a regular expression R, written as R^* is also a regular expression.

Ex: $\Sigma = \{a\}$ then a^* denotes the closure of Σ.

$a^* = \{\varepsilon, a, aa, aaa, \ldots\}$

Conventions on regular expressions

1. The operation '*' has highest precedence over concatenation, which has precedence over union (+).
   i.e., $RE\ (a + (b(c^*))) = a + bc^*$
2. The concatenation of K r's, where r is a regular expression is written as r^k. The language corresponding to r^k is L_r^k. Where L_r is language corresponding to regular expression r i.e., $rr = r^2$
3. r^+ is a regular expression to represent L_r^+

Note: A regular expression is not unique for a language i.e., regular language corresponds to more than one regular expression.

Example 15: Give regular expression for set of strings which either have 'a' followed by some b's or all b's also containing 'ε'.

(A) $b^* + ab^*$ (B) $a^* + ba^*$
(C) $(\varepsilon) + (\varepsilon + a)\ b^+$ (D) $b^* + ab^* + \varepsilon$

Solution: (C)

The regular expression is, $r = ab^+ + b^+ + \varepsilon = b^+(a + \varepsilon) + \varepsilon$.

Identity rules for regular expressions:

1. $\varnothing + R = R$
2. $\varnothing \cdot R = R\varnothing = \varnothing$
3. $\varepsilon R = R\varepsilon = R$
4. $\varnothing^* = \varepsilon$ and $\varepsilon^* = \varepsilon$
5. $R + R = R$
6. $RR^* = R^*\ R = R^+$
7. $\varepsilon + RR^* = R^*$ and $\varepsilon + R^*\ R = R^*$
8. $(R^*)^* = R^*$
9. $R^*\ R^* = R^*$
10. $\varepsilon + R^* = R^*$
11. $(R + \varepsilon)^* = R^*$
12. $R^*(\varepsilon + R)^* = (\varepsilon + R)^*\ R^* = R^*$
13. $R^*\ R + R = R^*\ R$
14. $(P + Q)R = PQ + QR$ and $R(P + Q) = RP + RQ$
15. $(P + Q)^* = (P^*\ Q^*)^* = (P^* + Q^*)^*$
16. $(PQ)^*\ P = P\ (QP)^*$
17. R is given as, $R = Q + RP$ has unique solution, $R = QP^*$. This is Arden's theorem.
18. $(P + Q)^* = (P^* + Q) = (P + Q^*)$

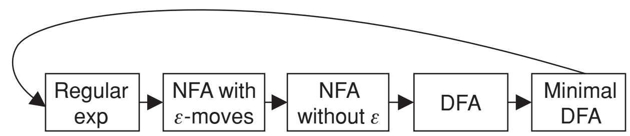
Example 16: If r_1 and r_2 are regular expressions denoting languages L_1 and L_2 respectively then which of following is false?

(A) $(r_1)|\ (r_2)$ is regular expression denoting $L_1 \cup L_2$.
(B) $(r_1)\ (r_2)$ is regular expression denoting $L_1 \cdot L_2$.
(C) $\varnothing$ is not a regular expression.
(D) $\{r_1\}^*$ is regular expression denoting L_1^*.

Solution: (C)

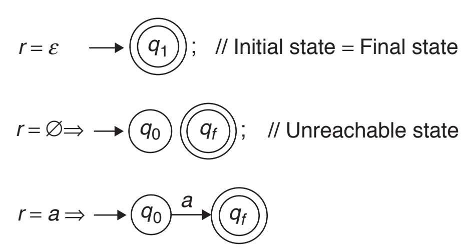
Constructing FA for Given RE

- Relationship between FA and RE.



Identities:

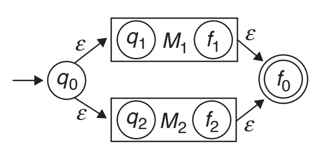
Basis:



Induction:

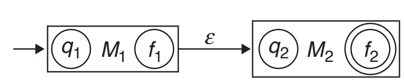
- **Union:** $L(r) = L\ (r_1) + L\ (r_2)$ i.e., $L\ (M) = L\ (M_1) \cup L\ (M_2)$

Let $M_1 = (Q_1, \Sigma_1, \delta_1, q_1, \{f_1\})$, $M_2 = (Q_2, \Sigma_2, \delta_2, q_2, \{f_2\})$ with $L\ (M_1) = L\ (r_1)$ and $L\ (M_2) = L\ (r_2)$, then $M = (Q_1 \cup Q_2 \cup \{q_0, f_0\}, \Sigma_1 \cup \Sigma_2, \delta, q_0, \{f_0\})$



- **Concatenation:**

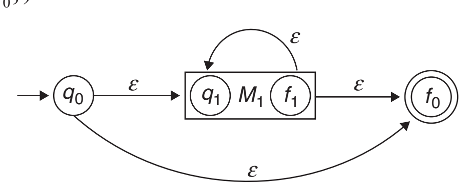
$L(r) = L\ (r_1) \cdot L\ (r_2)$ i.e., $L\ (M) = L\ (M_1) \cdot L\ (M_2)$



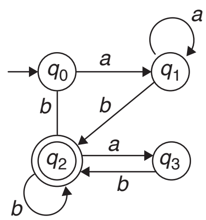
- **Closure:**

$L(r) = L(r)^*$ i.e., $L\ (M) = L\ (M_1)^*$

Let $M_1 = (Q_1, \Sigma_1, \delta_1, q_1, \{f_1\})$ then $L\ (M) = (Q_1 \cup \{q_0, f_0\}, \Sigma_1, \delta, q_0, \{f_0\})$



Example 17: The regular expression generated by the given FA.



(A) $(a + ba^*)\, b^*$ (B) $(aa^*b + bb^*)\, b^*$
(C) $(b + ab^*)\, a^*$ (D) $(ab + ba)^*$

Solution: (B)
q_2 is final state which is obtained with input symbol only 'b'. So, (C) or (D) is not true.

In (A) → ba^* is not defined in given *FA*. Instead bb^* is defined.

Pumping Lemma for Regular Sets

Theorem Let 'L' be an arbitrary regular language. Then there exists a positive integer, P with following property:

Given an arbitrary member, w of L having length at least P (i.e., $|w| \geq P$), w can be divided into 3-parts, $w = xyz\ \exists$

- $|y| \geq 1$ (the middle part is non-empty)
- $|xy| \leq P$ (the first two parts have length atmost P)
- For each, $i \geq 0$, $xy^iz \in L$ (removing or repeating the middle part produces member of L)

Proof Let L be an arbitrary regular language. Then there is a *FA*, say M that decides L.
Let P be the number of states of M.
Let w be an arbitrary member of L, having length 'n' with $n \geq P$.
Let $q_0, q_1, \ldots q_n$ be states that M on input w. That is, for each i, after reading the first i symbols of w, M is at q_i.
q_0 is initial state of M. Also, since $w \in L$, q_n is a final state of M.
Let $x = w_1 \ldots w_c$, $y = w_{c+4} \ldots w_d$, $z = w_{d+1} \ldots w_n$. Then:

- $|y| \geq 1$
- $|xy| \leq P$
- M transitions from q_0 to q_c on x.
- M transitions from q_c to q_c on y.
- M transitions from q_c to q_n on z.

Thus, for every $i \geq 0$, M transitions from q_0 to q_n on xy^iz and so, xy^iz is a member of L.

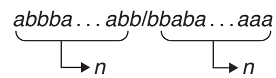
Note:

- Pumping lemma is used to verify that given language is not regular.
- Pumping lemma follows pigeon hole principle.

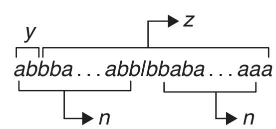
Example 18: The language, L is defined as:
$L = \{w_1w_2 : w_1, w_2 \in \{a, b\}^*, |w_1| = |w_2|\}$. Is the language regular?

(A) Regular
(B) Not regular
(C) Cannot be determined
(D) None of these

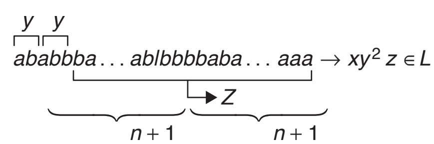
Solution: (A)
Fix pumping length, $K = 2$
For every proper strings in L, $(2n \geq 2)$



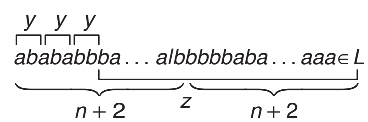
- Split in x, y, z with desired properties.



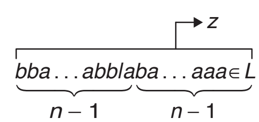
- Let $x = \varepsilon$, y = first two symbols, z = rest.



- xy^3z:



- xy^oz →



∴ For every $i \geq 0$, $xy^iz \in L$. Hence given language is regular.

Closure Properties of Regular Sets

1. **Union:** If L and M are regular languages, LUM is regular language closed under union.
2. **Concatenation and Kleen closure:** If L and M are regular languages, L.M is regular language and L^* is also regular.
3. **Intersection:** $L \cap M$ is regular, if L and M are regular languages.
4. **Difference:** $L - M$ contains strings in 'L' but not M, where L and M are regular languages.
5. **Complementation:** The complement of language L is $\Sigma^* – L$.

Note: Since Σ^* is surely regular, the complement of a regular language is always regular. Where Σ^* is a universal language.

6. **Homomorphism:** If L is a regular language, h is homomorphism on its alphabet then $h(L) = \{h(w) \mid w$ is in $L\}$ is also a regular language.

Regular grammar

- **Grammar:** Generative description of a language.
- **Automaton:** Analytical description.
- A grammar is a 4-tuple, G = (V, Σ, R, S) where V: alphabet (variable) (non-terminals)

$\Sigma \subseteq V$ is set of terminal symbols.
$R \subseteq (V^+ \times V^*)$ is a finite set of production rules.
$S \in V - \Sigma$ is start symbol.

Notation

- Elements of $V - \Sigma$: $A, B, \ldots$
- Elements of Σ: $a, b \ldots$
- Rules $(\alpha, \beta) \in R$: $\alpha \to \beta$ or $\alpha \xrightarrow[G]{} \beta$
- Start symbol is written as S.
- Empty word: ε

Example 19: The regular expression that describe the language generated by grammar, $G = (\{S, A, B\}, \{a, b\}, S, \{S \to Aab, A \to Aab|B, B \to a\}$

(A) $(ab)^* a$ (B) $aab(ab)^*$
(C) $ab^* aa$ (D) $(a + ba)^*$

Solution: (B)
$S \to Aab \to Aab\ ab \to A\ ab\ abab \to Bababab$
$\to aababab \to aab(ab)^*$

Union of two Regular languages:
If L_1 and L_2 are two languages then
$L_1 \cup L_2 = \{w/w \in L_1$ or $w \in L_2\}$
The union of two regular languages is also a regular language.
Let $M_1 = (Q_1, \Sigma, \delta_1, q_1, f_1)$
$M_2 = (Q_2, \Sigma, \delta_2, q_2, f_2)$
$M = M_1 U M_2$ can be given as
$M = (Q, \Sigma, \delta, q_0, f)$.
Where $Q = \{(r_1, r_2) \mid r_1 \in Q_1$ and $r_2 \in Q_2\}$
i.e., Q is the Cartesian product of sets Q_1 and Q_2.
Σ is the alphabet, is the same in M_1 and M_2.
$\Sigma = \Sigma_1 \cup \Sigma_2$.
δ is the transition function given as:
$\delta(r_1, r_2), a = (\delta_1(r_1, a)\ \delta_2(r_2, a))$.
q_0 is the pair (q_1, q_2).
F is the set of pairs in which either member is an accept state of M_1 or M_2.
$F = \{(r_1, r_2) \mid r_1 \in F_1$ or $r_2 \in F_2\}$

Types of Grammars

- Type 0: Unrestricted, recursively enumerable languages.
- Type 1: Context-sensitive grammar.
- Type 2: Context free grammar.
- Type 3: Regular grammar.

Type 0: Recursively enumerable grammar: (Turing Machine) (TM):

Every production rule is of form: $\alpha \to \beta$, where α and β are in $(V \cup T)^*$, i.e., there can be any strings of terminals and non-terminals (no-restriction).

Type 1: Context-sensitive Grammar: (Linear bounded automaton) (LBA):

Every production rule is of form, $\alpha \to \beta$ are in $(V \cup T)^*$ and $\alpha \neq \varepsilon$ and $|\beta| \geq |\alpha|$ i.e., any strings of terminals and non-terminals and length of string that can appear on RHS of production must be greater than or equal to length of string that can appear on LHS of production.

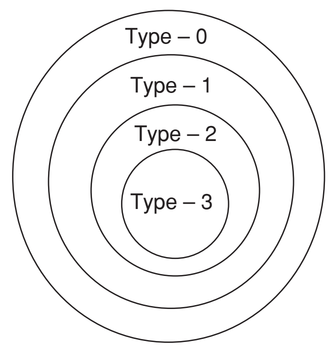
Type 2: Context-free grammar: (Push down automaton) (PDA):

Every production rule is of form, $A \to \alpha$ where α is in $(V \cup T)^*$ i.e., LHS of rule is single non-terminal and RHS can be any string of terminals and non-terminals.

Type 3: Regular grammar: (Finite automaton) (FA):

Every production is of form, $A \to aB$ or $A \to a$ where A and $B \in V$ and $a \in T$. That is, LHS of rule is non-terminal and RHS can be terminal (or) terminal followed by non-terminal.

Relationship between types of grammar:



- Regular sets are properly contained in CFL (Context Free Languages).
- The CFL's not containing empty string ε, are properly contained in CSL. (Context sensitive language).
- The CSL's are properly contained in Recursively enumerable languages.
- RG $\subset$ CFG $\subset$ CSL $\subset$ REG

Left-linear Grammar:
All productions have form: $A \to Bx$ or $A \to x$

Right-linear Grammar:
All productions have the form: $A \to xB$ or $A \to x$.

Note:

- The regular grammars characterize the regular sets i.e., a language is regular if and only if it has a left-linear grammar or if and only if it has a right-linear grammar.
- If L has a regular grammar, then L is a regular set.
- If L is a regular set, then L is generated by some left-linear grammar and by some right-linear grammar.

Arden's theorem: Let P and Q be two regular expressions over Σ. If P does not contain 'ε' then the following equation in R, namely $R = Q + RP$ has a unique solution given by $R = QP^*$.

Arden's Theorem to obtain regular expression from given transition diagram: The following steps are used to find the RE recognized by transition system.

The following assumptions are made regarding the transition system.

(i) The transition graph does not have ε-moves
(ii) It has only one initial state, q_o.
(iii) The states in the transition diagram are $q_o, q_1, q_2, \ldots q_n$.
(iv) Q_i, the regular expression represents the set of strings accepted by a system even though q_i is the final state.
(v) ${}^{a}ij$ denotes the regular expression representing the set of labels of edges from q_i to q_j. When there is no such edge ${}^{a}ij = \phi$.

We will get the following set of equations.

$$Q_1 = Q_1\alpha_{11} + Q_2\alpha_{12} + \ldots Q_n\alpha_{n1} + \varepsilon$$
$$Q_2 = Q_1\alpha_{12} + Q_2\alpha_{22} + \ldots Q_n\alpha_{n2}$$
$$\vdots$$
$$Q_n = Q_1\alpha 1n + Q_2\alpha_{2n} + \ldots Q_n\alpha_{nn}.$$

By Repeatedly applying substitutions and Arden's theorem, we can express Q_i in terms of α_{ij}'s.

For getting the set of strings recognized by the transition system, we have to take the union of all Q_i's corresponding to final states.

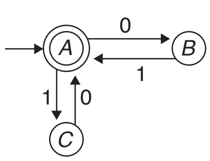
Construction of Regular Grammar from FA

Step I: Associate suitable variables like A, B, C … with states of automata.

Step II: Obtain the productions of the grammar as: If $\delta(A, a) = B$ then add production $A \to aB$ to list of productions of grammar, if B is a final state, then add either $A \to a$ or $B \to \varepsilon$, to list of productions of grammar.

Step III: The variable associated with initial state of automata is start symbol of grammar.

Example 20: Regular grammar generating language accepted by below automata is



(A) $A \to 0B|1C|\varepsilon$
$B \to 1A$
$C \to 0A$

(B) $A \to 1B|0C|\varepsilon$
$B \to 1A$
$C \to 0A$

(C) $A \to B|C|\varepsilon$
$B \to 1$
$C \to 0$

(D) $A \to 0A|1B|\varepsilon$
$B \to 1C$
$C \to 0A$

Solution: (A)
$A \to 0B$, $A \to 1C$, $B \to 1A$, $C \to 0A$
$\therefore$ A is final state, $A \to \varepsilon$

$\therefore A \to 0B\|1C\|\varepsilon$		$A \to 0B\|1C$
$B \to 1A$	(or)	$B \to 1A\|1$
$C \to 0A$		$C \to 0A\|0$

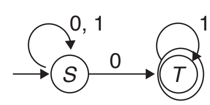
Construction of FA from given regular grammar

Given a regular grammar, G; a regular expression specifying $L(G)$ can be obtained directly as follows:

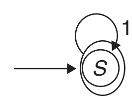
- Replace the '$\to$' symbol in productions of grammar by '=' symbol, to get set of equations.
- Solve the set of equations obtained above to get the value of variable, S, where S is start symbol of grammar, result is regular expression specifying $L(G)$.

Example 21: The Regular grammar and *FA* for given regular expression $\phi^* 1^* U\,(0\phi)^*$ is ___

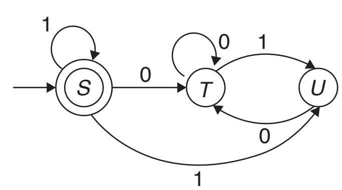
(A) $S \to 0S|1S|0$
$T \to 1T|\varepsilon$



(B) $S \to 1S|\varepsilon$

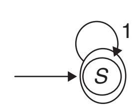


(C) $S \to 0T|1S|\varepsilon$
$T \to 0T|1U|\varepsilon$
$U \to 0T|1S$



(D) Cannot be determined

Solution: (B)
$\varnothing^*\ 1^* \cup (0\varnothing)^* = \varnothing^* \cdot 1^* \cup \varnothing^* = \varepsilon \cdot 1^* \cup \varepsilon = 1^*.$

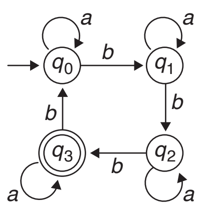


Exercises

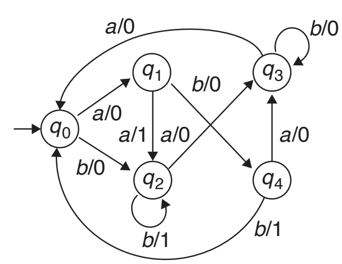
Practice Problems I

Directions for questions 1 to 15: Select the correct alternative from the given choices.

1. Find a regular expression for
   $L = \{uvu: u, v \in \{a, b\}^*, |u| = 2\}$
   (A) $(ab)^*a(ab)^*$
   (B) $(aa)^*ab(aa)^*$
   (C) $aa(a + b)^*bb + bb(a + b)^*aa$
   (D) $aa(a + b)^*aa + ab(a + b)^* ab + ba (a + b)^* ba + bb (a + b)^*bb$
2. Consider the regular expression, $R = 10 + (0 + 11)0^* 1$. The minimum number of states in any DFA accepting this regular expression is:
   (A) 5 (B) 4
   (C) 3 (D) 6
3. The following DFA accepts the set of all strings over $\{a, b\}$ that



   (A) Contains number of b's divisible by 3.
   (B) Contain number of a's and b's divisible by 3
   (C) Contain number of b's congruent to 3 modulo 4.
   (D) Contain any number of a's and b's
4. Consider the grammar, $S \to SS/a$. To get string of n terminals, the number of productions to be used is
   (A) n^2 (B) n
   (C) 2^{n+1} (D) $2n–1$
5. The language L is defined as, $L = \{a^i\, b^j\, c^{2j} |\ i \geq 0, j \geq 0\}$. Is this language L regular?
   (A) Yes (B) No
   (C) Cant be determined (D) None of these
6. The language, L is defined by set of strings over $\{a, b\}^*$ in which number of a's is a perfect cube. What is the nature of language, L?
   (A) Regular (B) Non-regular
   (C) Cant be determined (D) None of these
7. The language, L is defined over $\Sigma = \{0 – 7\}$. The string include 7, 16, 43, 61, 223, … The language generated is:
   (A) Alternate odd and even numbers
   (B) Octal representation of a number
   (C) Divisible by 7.
   (D) Octal representation of a number divisible by 7.
8. The language L, is defined as set of strings that start and end with equal number of a's and contain any number of b's. The grammar $L(G)$ for language L is defined with productions as:
   (A) $S \to aBa$
   $B \to \varepsilon|bB$
   (B) $S \to aB$
   $B \to a|bB$
   (C) $S \to aT|bS$
   $T \to aT|bT|a|b$
   (D) $S \to B|aSa$
   $B \to \varepsilon|bB$
9. If the regular set A is represented by $A = ((01)^*1^*)^*$. And the regular set B is represented by $B = (01 + 1)^*$, which of the following is true?
   (A) $A \subset B$
   (B) $B \subset A$
   (C) $A = B$
   (D) A and B are incomparable
10. The language, L that is generated over $\Sigma = \{0, 1\}$ for regular expression $L(r) = (0 + 10)^*\ 1\ (1 + 10)^*$
    (A) Any string whose number of 1's length is greater than or equal to 3.
    (B) Any string that has no substring 110.
    (C) Any string that has no substring 00 after first 11.
    (D) Any string that has only one occurrence of substring 010.
11. The R.E $L(r) = (a^+b^*)\ U\ \varepsilon$. Is the grammar with productions generated over non-terminals $\{S, A\}$ ambiguous?
    (A) Yes (B) No
    (C) Can't be determined (D) None
12. The number of states in the obtained Moore machine while converting the given mealy to Moore are:



    (A) 5 (B) 6
    (C) 4 (D) 7
13. The language L is defined as $L = \{0^i1^j / i \neq j\}$ over $\{0, 1, 2\}$, $A = \{0^i\ 1^j / i \geq 0, j \geq 0\}$ and $B = \{0^i\ 1^j / i = j\}$. For language, L to be non-regular. What should be relation between A, B, L?
    (A) $B = (A \cup L)^c$ (B) $B = A \cup L$
    (C) $B = A \cap \overline{L}$ (D) $B = A^c$
14. Which of following grammars are unambiguous?
    (A) $S \to (S)\ S|[S]\ S|\varepsilon$ (B) $S \to S(S)S|\varepsilon$
    (C) $S \to aS|Sa|a$ (D) $S \to a|Sa|bSS|Ssb|SbS$

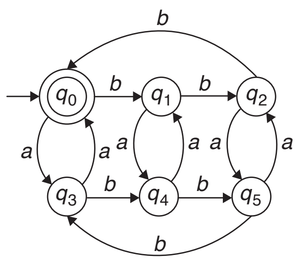
15. What will be number of final states obtained in DFA for language $L = \{w/w$ contains at least two 0's and atmost one 1$\}$ over $\Sigma = \{0, 1\}$.
(A) 2 (B) 1
(C) 3 (D) 4

Practice Problems 2

Directions for questions 1 to 15: Select the correct alternative from the given choices.

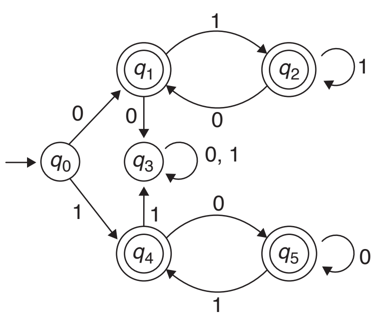
1. Which of the following Regular expression is equal to given regular expression: $(b + aa^* b) + (b + aa^*b)(a + ba^* b)^* (a + ba^* b)$
(A) $Ab (b + baa^*)$ (B) $a^*b (a + ba^* b)$
(C) $a^* b (a + ba^* b)^*$ (D) $ab (b + aa^*b)^*$

2. The following DFA accepts set of all strings over $\{a, b\}$ that contain



(A) Number of a's even and number of b's odd.
(B) Consecutive a's and b's
(C) Contain bbb as substring
(D) Number of a's even and number of b's divisible by three.

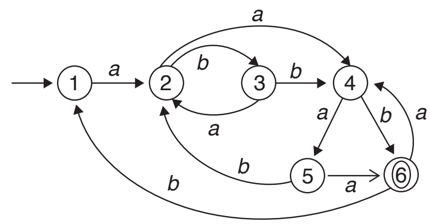
3. The regular language $L(r)$ for the given FSM is:



(A) It can start with zero followed by any number of 1's but no two consecutive 0's.
(B) It can start with 1, followed by any number of 0's but no two consecutive 1's.
(C) It is a combination of 0's and 1's but no two consecutive 0's or 1's.
(D) Both (A) and (B).

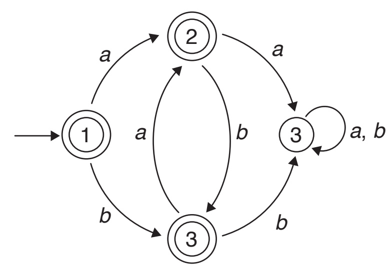
4. The language, L is defined as a set of non-palindromes over $\{a, b\}$. Is L regular?
(A) Yes (B) No
(C) Cannot be determined (D) None of above

5. The DFA, for language, L over $\Sigma = \{a, b\}$ is given below. What will be number of states in minimized DFA.



(A) 4 (B) 6
(C) 2 (D) 3

6. The minimal DFA given below is defined for language, $L = \{w \in \{a, b\}^*\}$ over $\Sigma = \{a, b\}$. The 'L' is:



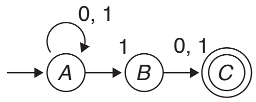
(A) Strings that contain equal number of a's and b's that have adjacent characters same.
(B) Contains adjacent characters same
(C) No two adjacent characters are same
(D) Starts and ends with same character that have adjacent character same.

7. The regular grammar $L(G)$ contains productions, P for language, $L = \{w \in \{a, b\}^*/$ there is at least one a$\}$ are:
(A) $S \to aS|bS|a|aT$
$T \to aT|bT|a|b$
(B) $S \to aS|bS|\varepsilon$
(C) $S \to aBb|bB$
$B \to a|b$
(D) $S \to bB$
$B \to b|\varepsilon$

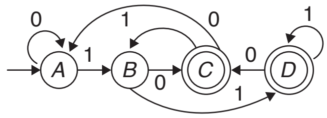
8. The regular expression for a language is defined as $((a^* b)^* (bc^*)^*)$. The total number of final states obtained in both NFA and DFA are respectively:
(A) 4, 2 (B) 1, 3
(C) 1, 5 (D) 2,3

9. The language, L is defined as $\{w/w$ has n occurrences of 0's where n mod 5 is 3$\}$ over $\Sigma = \{0, 1\}$. The number of final states obtained in the DFA for L is:
(A) 4 (B) 5
(C) 1 (D) 2

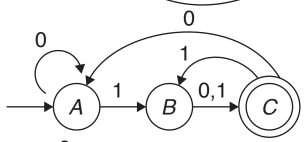
10. Which of the following is an equivalent DFA for the following NFA?



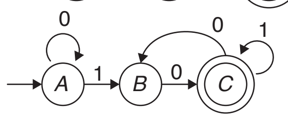
(A)



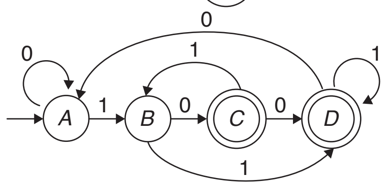
(B)



(C)



(D)



11. A regular grammar over alphabet $\Sigma = \{a, b, c, d\}$ whose language, is set of strings that contain exactly two b's is:

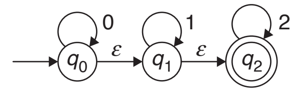
(A) $S \rightarrow aS|bS|cS|dA$
$A \rightarrow aA|bA|cA|dA|\varepsilon$

(B) $S \rightarrow aS|cS|dS|bB$
$B \rightarrow aB|cB|dB|bC$,
$C \rightarrow aC|cC|dC|\varepsilon$

(C) $S \rightarrow aS|bS|cS|dA$
$A \rightarrow aA|bB|cC$
$B \rightarrow b$
$C \rightarrow c$

(D) None of above

12. The following NFA contains ε-moves with 5, transitions. If this NFA with ε-moves is converted to NFA without ε-moves, what will be total number of transitions in obtained NFA?



(A) 5 (B) 4
(C) 6 (D) 3

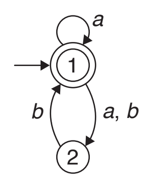
13. The regular expression, $r = (a + b)^*$. One more regular expression which represents same regular expression 'r' is:

(A) $a^* + b^*$ (B) $a^* \cdot b^*$
(C) $a^*(ba^*)^*$ (D) $(a + b)^* (a + b)$

14. The Regular grammar, $L(G)$ is defined for L with productions as $S \rightarrow Aab$, $A \rightarrow Aab|aB$, $B \rightarrow a$. What is Language generated by $L(G)$?

(A) Containing alternative a's and b's
(B) Containing alternative a's and b's, begins with an 'a' and ends with a 'b'.
(C) 'aa' followed by at least one set of alternating ab's.
(D) Consecutive aa's followed by 'b'.

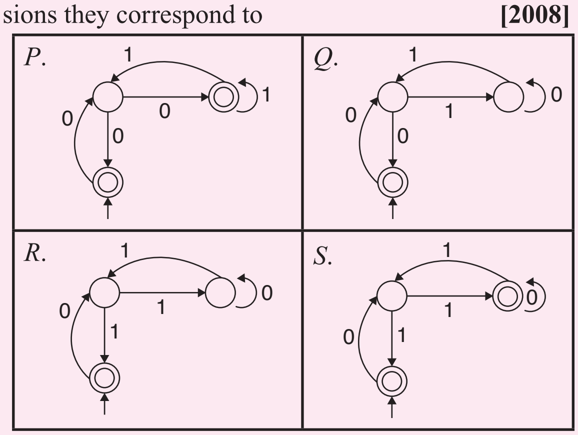
15. The number of final states in DFA after converting the NFA given below is:



(A) 4 (B) 2
(C) 3 (D) 1

Previous Years' Questions

1. Match the following NFAs with the regular expressions they correspond to **[2008]**



1. $\in + 0(01^*1 + 00)^*01^*$
2. $\in + 0(10^*1 + 00)^*0$
3. $\in + 0(10^*1 + 10)^*1$
4. $\in + 0(10^*1 + 10)^*10^*$

(A) P–2, Q–1, R–3, S–4
(B) P–1, Q–3, R–2, S–4
(C) P–1, Q–2, R–3, S–4
(D) P–3, Q–2, R–1, S–4

2. Which of the following are regular sets?

I. $\{a^n b^{2m} \mid n \geq 0, m \geq 0\}$
II. $\{a^n b^m \mid n = 2m\}$
III. $\{a^n b^m \mid n \neq m\}$
IV. $\{xcy \mid x, y \in \{a, b\}^*\}$ **[2008]**

(A) I and IV only (B) I and III only
(C) I only (D) IV only

3. Which one of the following languages over the alphabet $\{0, 1\}$ is described by the regular expression: $(0 + 1)^*0(0 + 1)^*0(0 + 1)^*$? **[2009]**

(A) The set of all strings containing the substring 00.
(B) The set of all strings containing atmost two 0's.
(C) The set of all strings containing at least two 0's.

(D) The set of all strings that begin and end with either 0 or 1.

4. Which one of the following is FALSE? **[2009]**

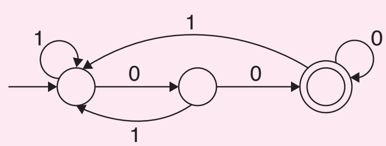
(A) There is a unique minimal DFA for every regular language.
(B) Every NFA can be converted to an equivalent PDA.
(C) Complement of every context-free language is recursive.
(D) Every non-deterministic PDA can be converted to an equivalent deterministic PDA.

5. Match all items in Group 1 with correct options from those given in Group 2. **[2009]**

Group 1		Group 2	
P.	Regular expression	1.	Syntax analysis
Q.	Pushdown automata	2.	Code generation
R.	Dataflow analysis	3.	Lexical analysis
S.	Register allocation	4.	Code optimization

(A) P–4, Q–1, R–2, S–3 (B) P–3, Q–1, R–4, S–2
(C) P–3, Q–4, R–1, S–2 (D) P–2, Q–1, R–4, S–3

6.



The above DFA accepts the set of all strings over {0, 1} that **[2009]**

(A) Begin either with 0 or 1
(B) End with 0
(C) End with 00
(D) Contain the substring 00.

7. Let $L = \{w \in (0 + 1)^* \mid w$ has even number of 1's$\}$, i.e., L is the set of all bit strings with even number of 1's. Which one of the regular expressions below represents L? **[2010]**

(A) $(0^*10^*1)^*$ (B) $0^*(10^*10^*)^*$
(C) $0^*(10^*1^*)^*0^*$ (D) $0^*1(10^*1)^*10^*$

8. Consider the languages $L_1 = \{0^i1^j \mid i \neq j\}$. $L_2 = \{0^i1^j \mid i = j\}$, $L_3 = \{0^i1^j \mid i = 2j + 1\}$. $L_4 = \{0^i1^j \mid i \neq 2j\}$. Which one of the following statements is true? **[2010]**

(A) Only L_2 is context free
(B) Only L_2 and L_3 are context free
(C) Only L_1 and L_2 are context free
(D) All are context free

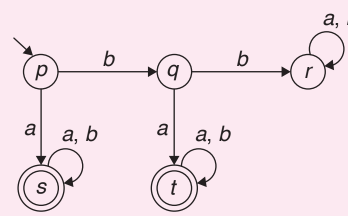
9. Let w be any string of length n in $\{0, 1\}^*$. Let L be the set of all substrings of w. What is the minimum number of states in a non-deterministic finite automaton that accepts L? **[2010]**

(A) n-1 (B) n
(C) n+1 (D) 2^{n-1}

10. Let P be a regular language and Q be a context-free language such that $Q \subseteq P$ (For example let P be the language represented by the regular expression p^*q^* and Q be $\{p^nq^n\}$ $n \in N\}$, Then which of the following is ALWAYS regular? **[2011]**

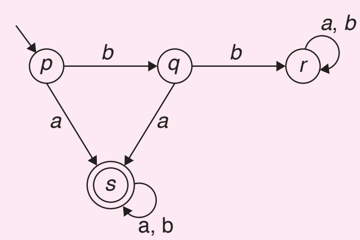
(A) $P \cap Q$ (B) $P - Q$
(C) $\Sigma^* - P$ (D) $\Sigma^* - Q$

11. A deterministic finite automaton (DFA) D with alphabet $\Sigma = \{a, b\}$ is given below.

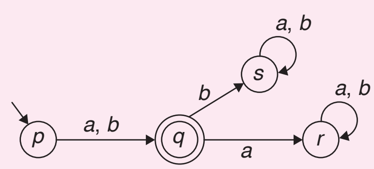


Which of the following finite state machines is a valid minimal DFA which accepts the same language as D? **[2011]**

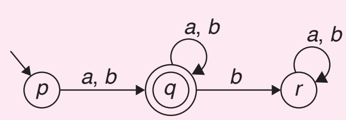
(A)



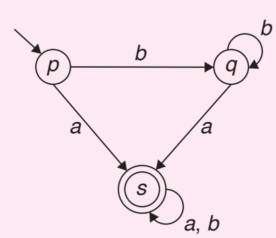
(B)



(C)



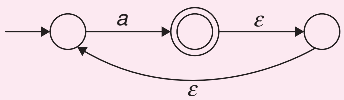
(D)



12. Given the language $L = \{ab, aa, baa\}$, which of the following strings are in L^*? **[2012]**

(1) *abaabaaabaa* (2) *aaaabaaaa*
(3) *baaaaabaaaab* (4) *baaaaabaa*
(A) 1, 2 and 3 (B) 2, 3 and 4
(C) 1, 2 and 4 (D) 1, 3 and 4

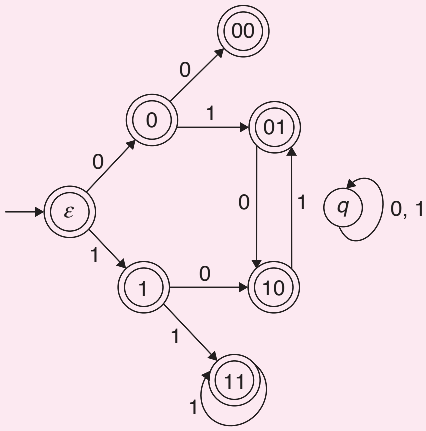
13. What is the complement of the language accepted by the NFA shown below?



Assume $\Sigma = \{a\}$ and ε is the empty string. **[2012]**

(A) $\varnothing$ (B) $\{\varepsilon\}$
(C) a^* (D) $\{a, \varepsilon\}$

14. Consider the set of strings on {0, 1} in which, every substring of 3 symbols has atmost two zeros. For example, 001110 and 011001 are in the language, but 100010 are not. All strings of length less than 3 are also in the language. A partially completed DFA that accepts this language is shown below.



The missing arcs in the DFA are **[2012]**

(A)

	00	01	10	11	q
00	1	0			
01				1	
10	0				
11			0		

(B)

	00	01	10	11	q
00		0			1
01		1			
10				0	
11		0			

(C)

	00	01	10	11	q
00		1			0
01		1			
10			0		
11		0			

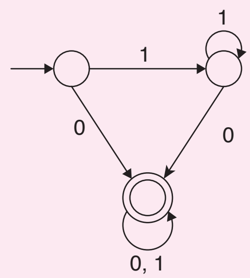
(D)

	00	01	10	11	q
00		1			0
01				1	
10	0				
11			0		

15. Consider the languages $L_1 = \Phi$ and $L_2 = \{a\}$. Which one of the following represents $L_1 L_2^* U L_1^*$? **[2013]**

(A) $\{\in\}$ (B) Φ
(C) a^* (D) $\{\in, a\}$

16. Consider the DFA A given below

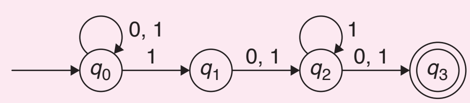


Which of the following are FALSE?

1. Complement of $L(A)$ is context-free.
2. $L(A) = L((11^*0 + 0)(0 + 1)^*0^*1^*)$
3. For the language accepted by A, A is the minimal DFA.
4. A accepts all strings over {0, 1} of length at least 2. **[2013]**

(A) 1 and 3 only (B) 2 and 4 only
(C) 2 and 3 only (D) 3 and 4 only

17. Consider the finite automaton in the following figure. **[2014]**



What is the set of reachable states for the input string 0011?

(A) $\{q_0, q_1, q_2\}$ (B) $\{q_0, q_1\}$
(C) $\{q_0, q_1, q_2, q_3\}$ (D) $\{q_3\}$

18. If $L_1 = \{a^n | n \geq 0\}$ and $L_2 = \{b^n | n \geq 0\}$, consider the statements **[2014]**

(I) $L_1 \cdot L_2$ is a regular language
(II) $L_1 \cdot L_2 = \{a^n b^n | n \geq 0\}$

Which one of the following is CORRECT?

(A) Only (I) (B) Only (II)
(C) Both (I) and (II) (D) Neither (I) nor (II)

19. Let $L_1 = \{w \in \{0, 1\}^* | w$ has at least as many occurrences of (110)'s as (011)'s}. Let $L_2 = \{w \in \{0, 1\}^* | w$ has at least

as many occurrences of (000)'s as (111)'s}. Which one of the following is TRUE? **[2014]**
(A) L_1 is regular but not L_2
(B) L_2 is regular but not L_1
(C) Both L_1 and L_2 are regular
(D) Neither L_1 nor L_2 are regular

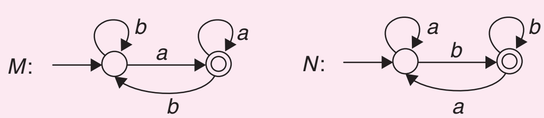
20. The length of the shortest string NOT in the language (over $\Sigma = \{a, b\}$) of the following regular expression is ______. **[2014]**
$a^*b^*(ba)^*a^*$

21. Let Σ be finite non-empty alphabet and let 2^{Σ^*} be the power set of Σ^*. Which one of the following is TRUE? **[2014]**
(A) Both 2^{Σ^*} and Σ^* are countable
(B) 2^{Σ^*} is countable and Σ^* is uncountable
(C) 2^{Σ^*} is uncountable and Σ^* is countable
(D) Both 2^{Σ^*} and Σ^* are uncountable

1. $\varepsilon + 0\,(01^*\,1 + 00)^*\,01^*$
2. $\varepsilon + 0\,(10^*\,1 + 00)^*\,0$
3. $\varepsilon + 0\,(10^*\,1 + 10)^*\,1$
4. $\varepsilon + 0\,(10^*\,1 + 10)^*\,10^*$

(A) $P - 2, Q - 1, R - 3, S - 4$
(B) $P - 1, Q - 3, R - 2, S - 4$
(C) $P - 1, Q - 2, R - 3, S - 4$
(D) $P - 3, Q - 2, R - 1, S - 4$

22. Consider the DFAs M and N given above. The number of states in a minimal DFA that accepts the language $L(M) \cap L(N)$ is _______ **[2015]**



23. The number of states in the minimal deterministic finite automaton corresponding to the regular expression $(0 + 1)^*(10)$ is ______ **[2015]**

24. Which of the following languages is/are regular? **[2015]**
L_1: $\{wxw^R | w, x \in \{a, b\}^*$ and $|w|, |x| > 0\}$, w^R is the reverse of string w
L_2: $\{a^nb^m | m \neq n$ and $m, n \geq 0\}$
L_3: $\{a^pb^qc^r | p, q, r \geq 0\}$
(A) L_1 and L_3 only
(B) L_2 only
(C) L_2 and L_3 only
(D) L_3 only

25. Consider the alphabet $\Sigma = \{0, 1\}$, the null/empty string λ and the sets of strings X_0, X_1 and X_2 generated by the corresponding non-terminals of a regular grammar. X_0, X_1 and X_2 are related as follows
$X_0 = 1X_1$
$X_1 = 0X_1 + 1X_2$
$X_2 = 0X_1 + \{\lambda\}$
Which one of the following choices precisely represents the strings in X_0? **[2015]**
(A) $10(0^* + (10)^*)1$
(B) $10(0^* + (10)^*)^*1$
(C) $1(0 + 10)^*1$
(D) $10(0 + 10)^*1 + 110(0 + 10)^*1$

26. Let L be the language represented by the regular expression $\Sigma^*\,0011\,\Sigma^*$ where $\Sigma = \{0, 1\}$. What is the minimum number of states in a DFA that recognizes $\bar{L}$ (complement of L)? **[2015]**
(A) 4
(B) 5
(C) 6
(D) 8

27. Which of the following languages is generated by the given grammar? **[2016]**
$S \rightarrow aS \mid bS \mid \varepsilon]$
(A) $\{a^n\,b^m \mid n, m \geq 0\}$
(B) $\{w \in \{a, b\}^* \mid$ w has equal number of a's and b's$\}$
(C) $\{a^n \mid n \geq 0\}\ U\ \{b^n \mid n \geq 0\}\ U\ \{a^nb^n \mid n \geq 0\}$
(D) $\{a, b\}^*$

28. Which of the following decision problems are undecidable? **[2016]**
I. Given NFAs N_1 and N_2, is $L(N_1) \cap L(N_2) = \Phi$?
II. Given a CFG $G = (N, \Sigma, P, S)$ and a string $x \in \Sigma^*$, does $x \in L(G)$?
III. Given CFGs G_1 and G_2, is $L(G_1) = L(G_2)$?
IV Given a TM M, is $L(M) = \Phi$?
(A) I and IV only
(B) II and III only
(C) III and IV only
(D) II and IV only

29. Which one of the following regular expressions represents the language: *the set of all binary strings having two consecutive 0's and two consecutive 1s?* **[2016]**
(A) $(0+1)^*\,0011\,(0+1)^* + (0+1)^*\,1100\,(0+1)^*$
(B) $(0+1)^*\,(00(0+1)^*11 + 11\,(0+1)^*00)\,(0+1)^*$
(C) $(0+1)^*\,00\,(0+1)^* + (0+1)^*\,11\,(0+1)^*$
(D) $00\,(0+1)^*\,11 + 11\,(0+1)^*\,00$

30. The number of states in the minimum sized **DFA** that accepts the language defined by the regular expression $(0+1)^*\,(0+1)\,(0+1)^*$ is ______ . **[2016]**

31. Language L_1 is defined by the grammar: $S_1 \rightarrow aS_1b \mid \in$
Language L_2 is defined by the grammar: $S_2 \rightarrow abS_2 \mid \in$

Consider the following statements:
P:L_1 is regular
Q:L_2 is regular
Which one of the following is TRUE? **[2016]**
(A) Both P and Q are true
(B) P is true and Q is false
(C) P is false and Q is true
(D) Both P and Q are false

32. Consider the following two statements:

I. If all states of an NFA are accepting states then the language accepted by the NFA is Σ^*.

II. There exists a regular language A such that for all languages B, A $\cap$ B is regular.

Which one of the following is **CORRECT**? **[2016]**
(A) Only **I** is true
(B) Only **II** is true
(C) Both **I** and **II** are true
(D) Both **I** and **II** are false

33. Consider the language L given by the regular expression $(a + b)^*b\ (a + b)$ over the alphabet $\{a, b\}$. The smallest number of states needed in a deterministic finite-state automaton (DFA) accepting L is ________. **[2017]**

34. The minimum possible number of states of a deterministic finite automaton that accepts the regular language $L = \{w_1aw_2 |\ w_1, w_2 \in \{a, b\}^*, |w_1| = 2, |w_2| \geq 3\}$ is ________. **[2017]**

35. Let δ denote the transition function and $\hat{\delta}$ denote the extended transition function of the $\in$-NFA whose transition table is given below:

δ	$\in$	a	b
$\to q_0$	$\{q_2\}$	$(q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\{q_2\}$	$\{q_3\}$
q_2	$\{q_0\}$	$\varnothing$	$\varnothing$
q_3	$\varnothing$	$\varnothing$	$(q_2\}$

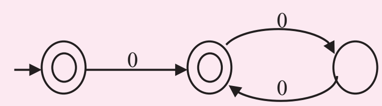
Then $\hat{\delta}\ (q_2, aba)$ is **[2017]**
(A) $\varnothing$
(B) $\{q_0, q_1, q_3\}$
(C) $\{q_0, q_1, q_2\}$
(D) $\{q_0, q_2, q_3\}$

36. Let N be an NFA with n states. Let k be the number of states of a minimal DFA which is equivalent to N. Which one of the following is necessarily true? **[2018]**
(A) $k \geq 2^n$
(B) $k \geq n$
(C) $k \leq n^2$
(D) $k \leq 2^n$

37. Given a language L, define L^i as follows:

$$L^0 = \{\varepsilon\}$$
$$L^i = L^{i-1}.\ L \text{ for all } i > 0$$

The order of a language L is defined as the smallest k such that $L^k = L^{k+1}$. Consider the language L_1 (over alphabet 0) accepted by the following automaton.



The order of L_1 is ______. **[2018]**

Answer Keys

Exercises

Practice Problems 1

1. D **2.** B **3.** C **4.** D **5.** B **6.** B **7.** D **8.** D **9.** C **10.** C
11. A **12.** D **13.** C **14.** A **15.** A

Practice Problems 2

1. C **2.** D **3.** D **4.** B **5.** B **6.** C **7.** A **8.** C **9.** C **10.** A
11. B **12.** C **13.** C **14.** C **15.** B

Previous Years' Questions

1. C **2.** A **3.** C **4.** D **5.** B **6.** C **7.** B **8.** D **9.** C **10.** C
11. A **12.** C **13.** B **14.** D **15.** A **16.** D **17.** A **18.** A **19.** A **20.** C
21. C **22.** 1 **23.** 3 **24.** A **25.** C **26.** B **27.** D **28.** C **29.** B **30.** 2
31. C **32.** B **33.** 4 **34.** 8 **35.** C **36.** D **37.** 2