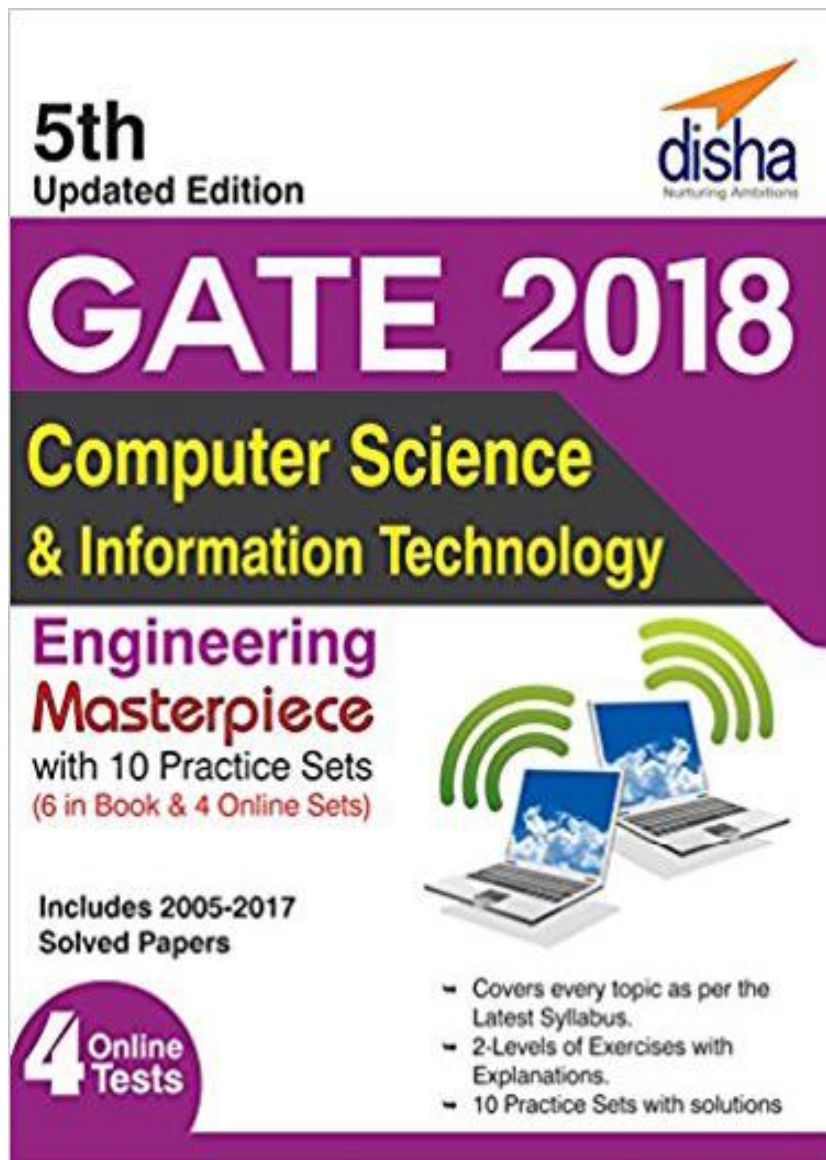


# Theory of Computation

*This Chapter “Theory of Computation” is taken from our:*



**ISBN : 9789386629029**

## THEORY OF COMPUTATION

### C O N T E N T S

- Regular languages and finite automata
- Context free languages and Push-down automata
- Recursively enumerable sets and Turing machines, Undecidability.

## ALPHABET AND STRINGS

An alphabet is a finite non-empty set of symbols. It is denoted by a letter  $\Sigma$ .

- $\Sigma = \{0, 1\}$  is a binary alphabet.
  - $\Sigma = \{A, B, \dots, Z; a, b, c, \dots, z\}$  is an english alphabet.
- A string over an alphabet  $\Sigma$  is a sequence of any number of symbols from  $\Sigma$ .
- 0, 1, 11, 00 and 011 are strings over  $\{0, 1\}$ .
  - Hey, Hello and Hi are strings over english alphabet.
- If  $w$  is a string over  $\Sigma$ , the length of  $w$ , written as  $|w|$  is the number of symbols that it contains
- Let  $\Sigma = \{a, b, \dots, z\}$ .
  - $|\text{hello}| = 5$ ;  $|\text{hey}| = 3$ ;  $|\text{hi}| = 2$ .
- The string of length zero is called the empty string and is written as  $\epsilon$ . In other words,  $\epsilon$  is a string containing no symbol.

## OPERATIONS ON STRINGS

### 1. Concatenation

Let string  $x$  of length  $a$  and string  $y$  of length  $b$ , the concatenation of  $x$  and  $y$ , written as  $xy$ , is the string obtained by appending  $y$  to the end of  $x$ , as in  $x_1 x_2, \dots, x_a y_1 y_2, \dots, y_b$ . and the length of concatenated string will be  $a + b$ .

- $\Sigma$  being English alphabet, Hello. John = Hello John.
- The concatenation of string  $x$  for  $n$  times, where  $n \geq 0$  is denoted by  $x^n$ .
- $x^0 = \epsilon$ ,  $x^1 = x$ ,  $x^2 = xx$ , .....

### 2. Substring

Let  $x$  and  $y$  be strings over an alphabet  $\Sigma$ . The string  $x$  is called substring of  $y$  if there exist strings  $m$  and  $n$  over  $\Sigma$  such that  $y = mxn$ ,  $m$  and  $n$  can be empty strings.  $\epsilon$  is a substring of every string and for every string  $x$ ,  $x$  is a substring of  $x$  itself.

- $\epsilon$ , concat, nation and concatenation are substring of concatenation.

### 3. Reverse

The reverse of string  $x$ , written as  $x^r$  is the string obtained by writing  $x$  in the opposite order.

- $(\text{hello})^r = \text{olleh}$ ;  $(\text{concatenation})^r = \text{noitanetacnoc}$
- The set of strings created from any number of symbols in an alphabet  $\Sigma$  is denoted by  $\Sigma^*$ .
- Let  $\Sigma = \{0, 1\}$ ,  $\Sigma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, \dots\}$ .
- The set of strings created from at least one symbol in an alphabet  $\Sigma$  is denoted by  $\Sigma^+$ .
- Let  $\Sigma = \{0, 1\}$ ,  $\Sigma^+ = \{0, 1, 00, 01, 10, 11, \dots\}$ .

## LANGUAGE

A language over an alphabet  $\Sigma$  is a set of strings over  $\Sigma$ . Let  $\Sigma = \{0, 1\}$  be the alphabet.

- $L_e = \{w \in \Sigma^* \mid \text{the number of 1's in } w \text{ is even}\}$
- The above language  $L_e$  contains all the strings over  $\Sigma$  having even number 1's.  $\epsilon, 0, 00, 11, 000, 101, 110, 1111, 0000, 1100, 0011, \dots$  are in  $L_e$ .
- BASIC operations like Complementation, Union, Intersection, Concatenation can be applied on languages. We will look into it afterwards.

## FINITE AUTOMATA

A finite automata is a 5-tuple  $(Q, \Sigma, \delta, q_0, F)$ , where

1.  $Q$  is a non-empty finite set called states.
2.  $\Sigma$  is a non-empty finite set called alphabet.
3.  $\delta : Q \times \Sigma \rightarrow Q$  is the transition function.
4.  $q_0 \in Q$  is the start state.
5.  $F \subseteq Q$  is the set of accept states.

242

Let's have a look at the diagram below, also called state diagram.

State diagram represents an automation say  $M$ .  $M$  is having three states, labelled  $q_1$ ,  $q_2$ ,  $q_3$  which is nothing but  $Q = \{q_1, q_2, q_3\}$ . The start state,  $q_1$  is indicated by the arrow pointing at it from nowhere. The accept state,  $q_2$ , is the one with a double circle. The arrows going from one state to another are called transitions. When this automation receives an input string, it processes that string and produces an output. The output is either accept or reject. If after reading the last symbol of string, automation is at accept state then the output is accept, otherwise rejected.

For example, when the input is feeded at string 1101 to the machine  $M$  shown above, the processing proceeds as follows:-

1. Start in state  $q_1$ .
2. Read 1, follow transition from  $q_1$  to  $q_2$ .
3. Read 1, follow transition from  $q_2$  to  $q_2$ .
4. Read 0, follow transition from  $q_2$  to  $q_3$ .
5. Read 1, follow transition from  $q_3$  to  $q_2$ .
6. ACCEPT because  $M$  is in accept state  $q_2$  at the end.

Formally the above machine can be written as  $M = (Q, \Sigma, \delta, q_1, F)$  where,

→  $Q = \{q_1, q_2, q_3\}$

→  $\Sigma = \{0, 1\}$

→  $\delta$  is described as

→  $q_1$  is the start state, and

→  $F = \{q_2\}$

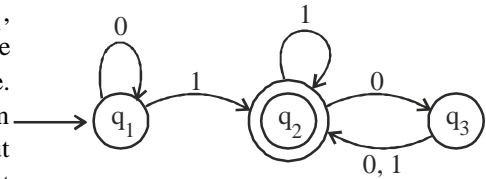
	0	1
$q_1$	$q_1$	$q_2$
$q_2$	$q_3$	$q_2$
$q_3$	$q_2$	$q_2$

- If  $A$  is the set of all strings that machine  $M$  accepts, we say that  $A$  is the language of machine  $M$  and write  $L(M) = A$ . We say the  $M$  recognizes  $A$  or that  $M$  accepts  $A$ .
- A machine may accept several strings, but it always recognizes only one language. If the machine accepts no strings, it still recognizes one language called empty language  $\phi$ .
- In the previous example,  
 $A = \{w | w \text{ contains at least one 1 and an even number of 0 s follow the last 1}\}$ .
- A language is called regular language if some finite automation recognizes it. Language  $A$  defined above is a regular language because  $M$  recognizes  $A$ .

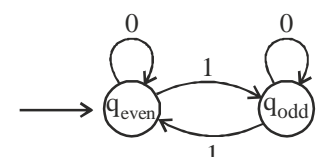
### Designing Finite Automata (DFA)

Let's look at a simple example to understand how to design an automation. Suppose that the alphabet is  $\{0, 1\}$  and that the language consists of all strings with an odd number of 1's. We have to construct a finite automation  $M$  to recognize this language. To construct the automation for this language, remember one thing that whether the number of 1's seen so far is even or odd and keep track of this information as read new symbols. If you read 1, flip the answer, but if you read a 0, leave the answer as it is. After this, we evaluate what are the finite list of possibilities. Here there are two. One is when number of 1's are even and other when they are odd. Now we assign a state to each of the possibilities, say  $q_{\text{even}}$  and  $q_{\text{odd}}$ . Next you need to assign the transitions by seeing how to go from one possibility to another upon reading a symbol. So, if state  $q_{\text{even}}$  represent the even possibility and  $q_{\text{odd}}$  represent odd possibility, you would set the transition to flip state on a 1 and stay put on 0. Finally, start state will be the state  $q_{\text{even}}$  because initially there are 0 number of 1's and 0 is an even number. Accept state will be  $q_{\text{odd}}$  because you have to accept when you have seen an odd number of 1s. In figure is the corresponding automata.

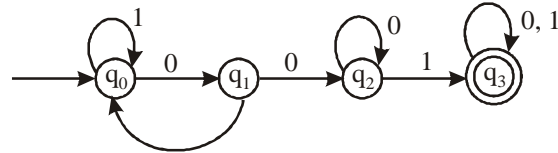
**Note:** In GATE examination, you will be asked either to determine the string accepted by automation or the correct design of automation for a given string.



State diagram



**Example 1.** Consider a following deterministic finite state automaton  $M$



Let  $S$  denotes the set of seven bit binary string  $S$  in which the first, the fourth and the last bits are 1. The number of string in  $S$  that are accepted by  $M$  is

- (a) 1 (b) 5  
(c) 7 (d) 8

**Sol. (c)** The string  $S$  will be defined as

1 x y 1 z  $\omega$  1 (7 digits)

Here, the values of  $x$ ,  $y$ ,  $z$  and  $\omega$  can be either 0 or 1.

As shown in figure the machine will jump to next state only after the receiving input 0 or 1. So, the next value will be 0 thus  $S$  is fixed to

(replacing  $x$ ) 10 y 1 z  $\omega$  1 (State  $q_1$ )

or (replacing  $x$ ) 11 y 1 z  $\omega$  1 (state  $q_0$ )

For sequence 10 y z  $\omega$  1 the machine is on state  $q_2$ . Thus, we have two values of  $S$  that is

(replacing  $y$ ) 1001 z  $\omega$  1 (State  $q_3$ )

or (replacing  $y$ ) 1011 z  $\omega$  1 (State  $q_0$ )

For string 1001 z  $\omega$  1 the string will be

(replacing  $y$ ) 1101 z  $\omega$  1 (State  $q_0$ )

or (replacing  $y$ ) 1111 z  $\omega$  1 (State  $q_0$ )

For string 1001 z  $\omega$  (State  $q_3$ ) the machine is on state  $q_3$ . Thus, the next string will be

(replacing  $z$ ) 10010  $\omega$  1 (State  $q_3$ )

(replacing  $z$ ) 10011  $\omega$  1 (State  $q_3$ )

For string 1011 z  $\omega$  1 (State  $q_0$ )

The value of  $z$  can be only 0 otherwise the machine will not reach to final state thus the string  $S$  will be

(replacing  $z$ ) 10110  $\omega$  1 (State  $q_1$ )

In this case the value of  $\omega$  will also be 0 to jump to state  $q_3$  so that on receiving 1 to machine could jump on final state. Thus the string will be

(replacing  $\omega$ ) 1011001S.

For string 1101 z  $\omega$  1 (state  $q_0$ ) the value of  $z$  will be 0 and  $\omega$  will also be 0.

So that the machine could reach to (replacing  $z$ ,  $\omega$ ) 1101001S

and 1111001S (replacing  $z$ ,  $\omega$ ) for string 100 10 $\omega$  (state  $q_3$ ) the value of  $w$

will be so that the machine could reach final state. Thus, the string will be (replacing  $z$ ,  $\omega$ ) 1001011 S

For string 10011  $\omega$  1 (State  $q_3$ ) the machine is on state  $q_3$ . So, it is on final state and the input  $\omega$  can be either 0 or 1. As both transactions are available on final state. Thus, the string can be

or 1001101S

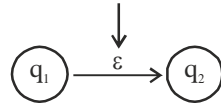
1001111S

### Non-Deterministic Finite Automata (NFA)

So far, every step of a computation follows in a unique way from the preceding step. When the machine is in a given state and reads the next input symbol, we know the next state (using transition), i.e. it is determined. This is called deterministic computation. In a non-deterministic machine, several choices may exist for the next state at any

244

point. Also, the automation can choose to make a transition from one state to another without reading any symbol. This type of transition is shown below.



Formally, a non-deterministic finite automation is a 5-tuple  $(Q, \Sigma, \delta, q_0, F)$ , where

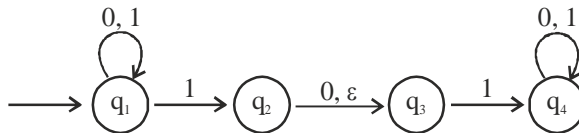
1.  $Q$  is a non-empty finite set of states.
2.  $\Sigma$  is a non-empty finite alphabet.
3.  $\delta: Q \times \Sigma_{\epsilon} \rightarrow P(Q)$  is the transition function. Also,  $Q \times \Sigma \Rightarrow 2^Q$ .
4.  $q_0 \in Q$  is the start state.
5.  $F \subseteq Q$  is the set of accept states.

For example, NFA drawn below can be formally written as:-

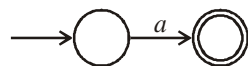
1.  $Q = \{q_1, q_2, q_3, q_4\}$
2.  $\Sigma = \{0, 1\}$
3.  $\delta$  is given as

	0	1	$\epsilon$
$q_1$	$\{q_1\}$	$\{q_1, q_2\}$	$\phi$
$q_2$	$\{q_3\}$	$\phi$	$\{q_3\}$
$q_3$	$\phi$	$\{q_4\}$	$\phi$
$q_4$	$\{q_4\}$	$\{q_4\}$	$\phi$

4.  $q_1$  is the start state.
5.  $F = \{q_4\}$



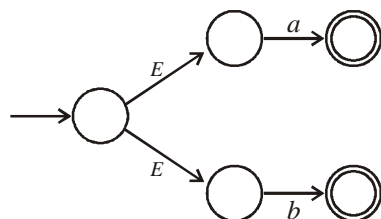
Step of type (i):  $a$



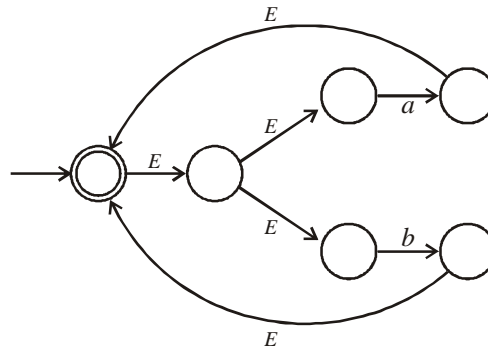
Step of type (i):  $b$



Step of type (ii):  $a/b$



Step of type (iv):  $(a/b)^*$



Step of type (iii):  $(a/b)^*a$

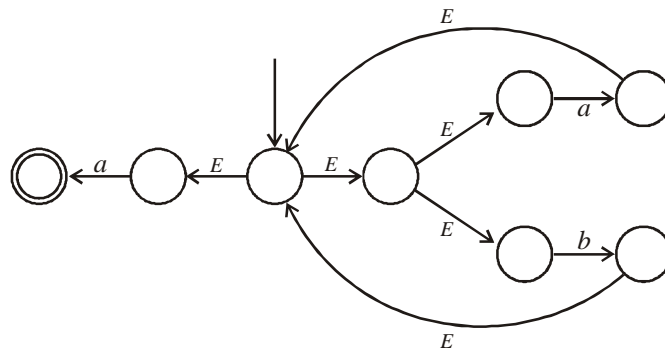


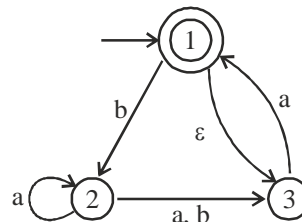
Figure 1 : Steps in constructing an  $NFA^E$  for  $(a/b)^*a$

**Some important points to remember:**

1. Any DFA is also an NFA.
2. For each NFA, there exist an equivalent DFA.
3. Taking into account above two points, a language is regular if and only if some NFA recognizes it.
4. If  $n$  is the number of states in NFA, then the equivalent DFA will have  $2^n$  state including  $\phi$  state.

**Converting NFA to DFA**

Let's have a look at the NFA(N) below that we will be converting in DFA(D).



To construct a DFA(D) that is equivalent to N, we first need to determine D's states. Since N is having 3 states, D's states will be  $2^3 = 8$ . Now label each of D's state with the subset of N's state. Thus D's state set is

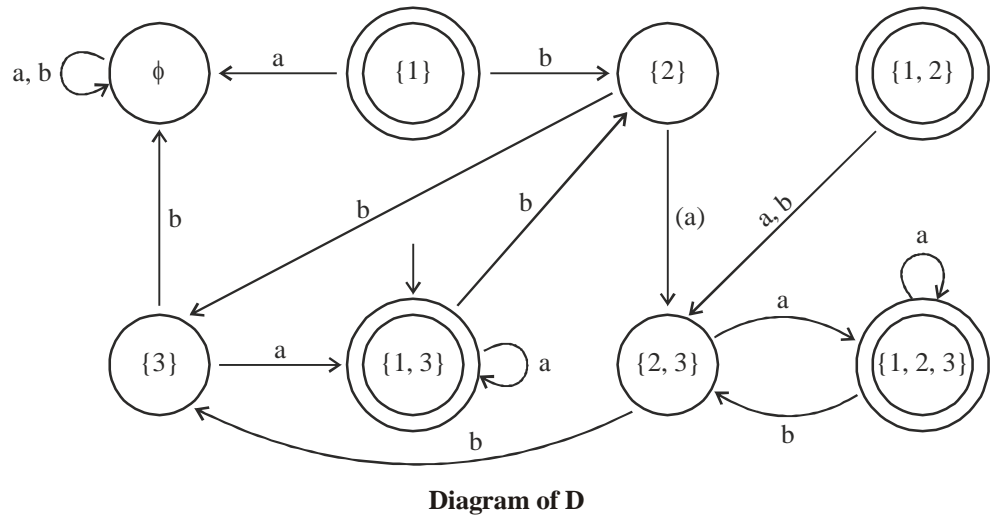
$$\{\phi, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}.$$

The start state of D is the state that are reachable from 1 by travelling along  $\epsilon$  arrows. An  $\epsilon$  arrow goes from 1 to 3, so start state is  $\{1, 3\}$ . Accept states of D are the states containing N's accept state, i.e.  $\{\{1\}, \{1, 2\}, \{1, 3\}, \{1, 2, 3\}\}$ . Now, we determine D's

246

transition function. In D, state {2} goes to {2, 3} on input a, because in N, state Z goes to both 2 and 3 on input a and we can't go farther from 2 or 3 along  $\epsilon$  arrows. State {2} goes to {3} on input b. Similarly, state {1} goes to  $\phi$  on input a, because no a arrows exit it in N. Continuing in this way, we obtain the diagram of D.

The above DFA can be simplified by removing the unwanted states. In this case, they are {1} and {1, 2} because there are no arrows pointing at these states and also none of them are start state and hence they can't be reached. So, the simplified DFA will be



## FINITE AUTOMATA WITH OUTPUTS

Under finite automata we have two other machines. (a) Mealy machine and (b) Moore machine. These machines produces some output also at every state.

### (i) Mealy Machine

A mealy machine is a tuple

$(Q, \Sigma, \Delta, \delta, \lambda, q_0)$  where

$Q$  = finite set of states

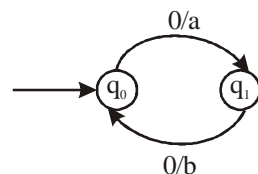
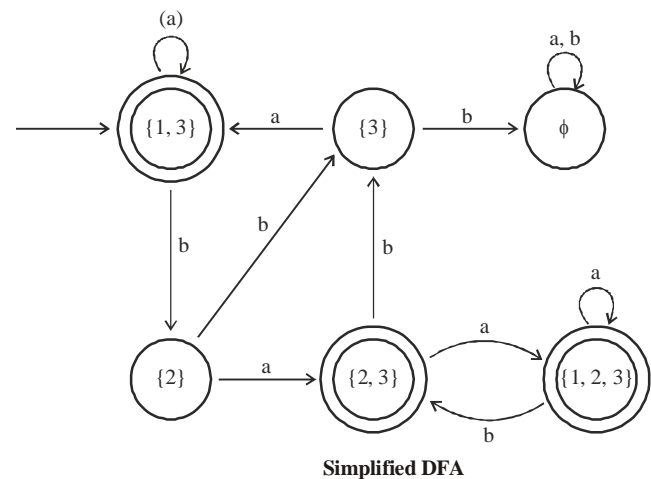
$\Sigma$  = finite non-empty set of input alphabets

$\Delta$  = the set of output alphabets

$\delta$  = transition function defined as  $Q \times \Sigma \rightarrow Q$ .

$\lambda$  = the output function defined as  $\Sigma \times Q \rightarrow \Delta$  means at any state the value of current state and current input defines the output

$q_0$  = initial state



Here, in the given mealy machine when input 0 is applied to state  $q_0$  it produces output a and when 0 is applied to state  $q_1$  it produces the output b. The transition table will be represented as

State	Input	0
	Next state	Output
$\rightarrow q_0$	$q_1$	a
$q_1$	$q_0$	b

In case of Mealy machine output is ^ if input is ^.

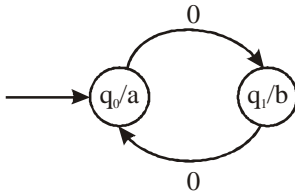
### (ii) Moore Machine

Moore machine is a set of six tuples  $(Q, \Sigma, \Delta, \lambda, q_0)$  where,

$Q$  = set of states

$\Sigma$  = set of inputs





$\Delta$  = set of outputs

$\delta$  = transition function defined as  $Q \times \Sigma \rightarrow Q$

$\lambda$  = output function defined as  $Q \rightarrow \Delta$  means only the current state of automata will define the output

$q_0$  = the initial state

As we can illustrate the Moore machine in following example.

Herre,  $q_0$  state will generate output a and  $q_1$  state will generate output b. The transition table will be given as

State	Input = 0	Output
$q_0$	$q_1$	a
$q_1$	$q_0$	b

### Finite automata from regular expressions

Given a regular expression  $r$ , over an alphabet  $\Sigma$  say, we wish to construct a DFA  $M$  with alphabet of input symbol  $\Sigma$  and with the property tht for each  $u \in \Sigma^*$ ,  $u$  matches  $r$  iff  $u$  is accepted by  $M$ —so that  $L(r) = L(M)$ .

Note that by the Theorem on Slide 18 it is enough to construct an  $NFA^E N$  with the property  $L(N) = L(r)$ . For then we can apply the subset construction to  $N$  to obtain a DFA  $M = PN$  with  $L(M) = L(PN) = L(N) = L(r)$ . Working with finite automata that are non-deterministic and have  $\epsilon$ -transitions simplifies that construction of a suitable finite automaton from  $r$ .

Let us fix on a particular alphabet  $\Sigma$  and from now on only consider finite automata whose set of input symbols is  $\Sigma$ . The construction of an  $NFA^E$  for each regular expression  $r$  over  $\Sigma$  proceeds by recursion on the syntactic structure of the regular expression, as follows.

- For each atomic form of regular expression,  $a$  ( $a \in \Sigma$ ),  $\epsilon$ , and  $\emptyset$ , we give an  $NFA^E$  accepting just the strings matching that regular expression.
- Given any  $NFA^E$ s  $M_1$  and  $M_2$ , we construct a new  $NFA^E$ , *Union* ( $M_1, M_2$ ) with the property

$$L(\text{Union}(M_1, M_2)) = \{u \mid u \in L(M_1) \text{ or } u \in L(M_2)\}.$$

Thus  $L(r_1|r_2) = L(\text{Union}(M_1, M_2))$  when  $L(r_1) = L(M_1)$  and  $L(r_2) = L(M_2)$ .

- Given any  $NFA^E$ s  $M_1$  and  $M_2$ , we construct a new  $NFA^E$ , *Concat* ( $M_1, M_2$ ) with the property.

$$L(\text{Concat}(M_1, M_2)) = \{u_1u_2 \mid u_1 \in L(M_1) \text{ and } u_2 \in L(M_2)\}.$$

Thus  $L(r_1r_2) = L(\text{Concat}(M_1, M_2))$  when  $L(r_1) = L(M_1)$  and  $L(r_2) = L(M_2)$ .

- Given any  $NFA^E M$ , we construct a new  $NFA^E$ , *Star* ( $M$ ) with the property

$$L(\text{Star}(M)) = \{u_1u_2...u_n \mid n \geq 0 \text{ and each } u_i \in L(M)\}.$$

Thus  $L(r^*) = L(\text{Star}(M))$  when  $L(r) = L(M)$ .

Thus starting with step (i) and applying the constructions in steps (ii)-(iv) over and over again, we eventually build  $NFA^E$ s with the required property for every regular expression  $r$ .

Put more formally, one can prove the statement

for all  $n \geq 0$ , and for all regular expressions of size  $\leq n$ , there exists an  $NFA^E M$  such that  $L(r) = L(M)$

by mathematical induction on  $n$ , using step (i) for the base case and steps (ii)-

- for the induction steps. Here we can take the *size* of a regular expression to be the number of occurrences of union ( $- / -$ ), concatenation ( $- -$ ), or star ( $-^*$ ) in it.

**Example 2.** Convert the following Mealy machine to Moore machine.

State	Input			
	0		1	
	Next state	Output	Next state	Output
$q_1$	$q_3$	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

**Sol.** Here, if any state have same output for all input values like in our example,  $q_1$  state have output 1 only for both 0 and 1 input. thus, the state will be written as it is. If state has different output for different input like for state  $q_2$  for input 0 output is 1 and for input 1 output is 0. In this case state  $q_2$  will be divided into 2 parts. Where it is total number of inputs. Here  $q_2$  will be divided into two parts  $q_{20}$  and  $q_{21}$ .  $q_{20}$  will define the output 0 and  $q_{21}$  will define the output 1. Same thing will be done for state  $q_4$ . Thus, the transition table will be written as

State	0	1	Output
$q_1$	$q_3$	$q_2$	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

**Example 3.** Convert the following Moore machine into Mealy machine.

State	0	1	Output
$q_1$	$q_1$	$q_2$	0
$q_2$	$q_1$	$q_3$	1
$q_3$	$q_1$	$q_3$	1

**Sol.** The conversion is simple. Copy the output associated with each state to the place where state is written. Thus, the example output of  $q_1$  state is 0.

Thus,  $q_1 \times 0$  will generate the output 0.

And output of  $q_2$  state is then the transition

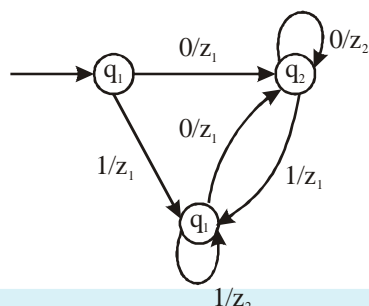
$q_1 \times 1 \rightarrow q_2$  will generate the output 1.

$q_1 \times \rightarrow 1$

And the new transition table will be

State	0		1	
	Next state	Output	Next state	Output
$q_1$	$q_1$	0	$q_2$	1
$q_2$	$q_1$	0	$q_3$	1
$q_3$	$q_1$	0	$q_3$	1

**Example 4.** Convert the following Mealy machine into Moore machine.

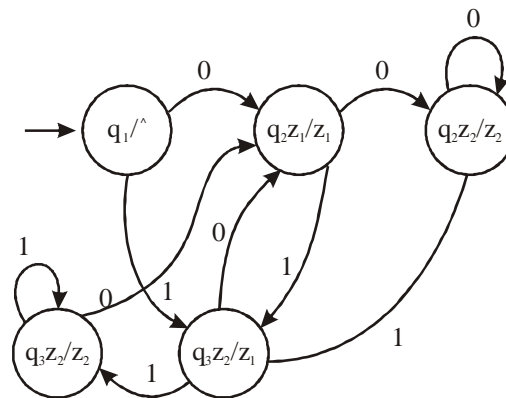


**Sol.** The transition table of given Mealy machine will be as

State	0	1
	Next state    Output	Next state    Output
$q_1$	$q_2$ $z_1$	$q_3$ $z_1$
$q_2$	$q_2$ $z_2$	$q_3$ $z_1$
$q_3$	$q_2$ $z_1$	$q_3$ $z_2$

The corresponding Moore machine will be

	0	1	Output
$q_1$	$q_2 z_1$	$q_2 z_1$	$z_1$
$q_2 z_1$	$q_2 z_2$	$q_3 z_1$	$z_1$
$q_2 z_2$	$q_2 z_2$	$q_3 z_1$	$z_2$
$q_3 z_1$	$q_2 z_1$	$q_3 z_2$	$z_1$
$q_3 z_2$	$q_2 z_1$	$q_3 z_2$	$z_2$



**Example 5.** Construct a minimum automaton equivalent to finite automaton described by

State	0	1
$\rightarrow q_0$	$q_1$	$q_5$
$q_1$	$q_6$	$q_2$
$\odot q_2$	$q_0$	$q_2$
$q_3$	$q_2$	$q_6$
$q_4$	$q_7$	$q_5$
$q_5$	$q_2$	$q_6$
$q_6$	$q_6$	$q_4$
$q_7$	$q_6$	$q_2$

**Sol.** **Step 1** Make separate set of final states and non-final states, i.e.,

$\{q_0, q_1, q_3, q_4, q_5, q_6, q_7\} \{q_2\}$

**Step 2** Further in a given set check whether for two given state, the next state for a given input belongs to same set. If yes then keep these two states in same set. Other wise, these two state will be in different set. For example, states  $q_0$  and  $q_1$  form set 1. State  $q_1$  for input 0 and  $q_1$  generates  $q_6$  for input 0.  $q_1$  and  $q_6$  belongs to same set. Now for input 1  $q_0$  generates  $q_5$  and  $q_1$  generates  $q_2$ .  $q_5$  and  $q_2$  are in different set.  $q_0$  and  $q_1$  will be in different set.

Here, notice that  $q_1$  generates  $q_1$  for input 0 and  $q_5$  for input 1.  $q_1$  and  $q_5$  both are intermediate states and belongs to set 1. Thus, all states that generate intermediate states for both inputs 0 and 1 will be in set of  $q_0$  thus we will have the 1 set as

$\{q_0, q_4, q_6\}$

Next  $q_1$  generates state  $q_6$  for input 0 which is intermediate state and generates state  $q_2$  for input 1 which intermediate state for input 0 and any final state for input 1 will be set of  $q_1$ , thus we will have a set

$\{q_1, q_7\}$

Similarly,  $q_3$  generates  $q_2$  for input 0 which belongs to set 2 and generate  $q_6$

for input 1 which belongs to set 1. Thus, all states that generate any state that belongs to set 2 for input '0' and any state that belongs to set 1 for input 1 will be in set of  $q_3$ . Thus, we get set as

$$\{q_3, q_5\}$$

Thus, in next step we will get the set as

$$\{q_0, q_4, q_6\}, \{q_1, q_7\}, \{q_3, q_5\}, \{q_2\} \Rightarrow Q_2'$$

Applying the previously described process of this newly generated set  $Q_2'$ , we get

$$\{q_0, q_4\} \{q_6\} \{q_1, q_7\} \{q_3, q_5\}, \{q_2\} \Rightarrow Q_3'$$

By applying the same process, we get

$$\{q_0, q_4\} \{q_6\} \{q_1, q_7\} \{q_3, q_5\} \{q_2\} \Rightarrow Q_4'$$

$$Q_3' = Q_4'$$

Thus,  $Q' = [\{q_0, q_4\} \{q_6\} \{q_1, q_7\} \{q_3, q_5\} \{q_2\}]$

The transaction table will be

State	0	1
$\rightarrow \{q_0, q_4\}$	$\{q_1, q_7\}$	$\{q_3, q_5\}$
$\{q_6\}$	$\{q_6\}$	$\{q_2\}$
$\{q_1, q_6\}$	$\{q_6\}$	$\{q_2\}$
$\{q_3, q_5\}$	$\{q_2\}$	$\{q_6\}$
$\{q_2\}$	$\{q_0, q_4\}$	$\{q_2\}$

Here, the state  $q_0$  or  $q_4$  will be represented as  $\{q_0, q_4\}$  the state  $q_3$  or  $q_5$  will be represented as  $\{q_3, q_5\}$  and state  $q_1$  or  $q_7$  or  $q_5$  will be represented as  $\{q_3, q_5\}$  and state containing  $q_0$  will be represented as  $\{q_1, q_7\}$  the state containing  $q_0$  will be considered as initial state and the state containing any final state will be considered as final state.

## CLOSURE PROPERTIES OF REGULAR LANGUAGES

Regular languages are closed under the following operations:-

1. Union  $\rightarrow$  If  $A_1$  and  $A_2$  are regular, then  $A_1 \cup A_2$  is also regular.
2. Intersection  $\rightarrow$  If  $A_1$  and  $A_2$  are regular, then  $A_1 \cap A_2$  is also regular.
3. Concatenation  $\rightarrow$  If  $A_1$  and  $A_2$  are regular, then  $A_1 \circ A_2$  is also regular.
4. Star  $\rightarrow$  If  $A$  is regular language, then  $A^*$  is also regular.

## REGULAR EXPRESSION

The regular expressions are useful for representing certain sets of string in an algebraic fashion. Formal definition of regular expressions is as follows:-

1. Any terminal symbol, i.e. symbols  $\Sigma$  including  $\wedge$  and  $\phi$  are regular expression. When, we view  $a$  in  $\Sigma$  as  $a$  regular expression, we denote it by  $\underline{a}$ .
2. The union of two regular expressions  $R_1$  and  $R_2$  written as  $R_1 + R_2$  also a regular expression.
3. The concatenation of two regular expressions  $R_1$  and  $R_2$  written as  $R_1 R_2$  is also a regular expression.
4. The iteration of a regular expression  $R$ , written as  $R^*$  is also a regular expression.
5. If  $R$  is a regular expression, then  $(R)$  is also a regular expression.
6. The regular expression over  $\Sigma$  are precisely those obtained successively by the application of the rules 1 to 5 once or several times.

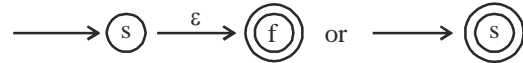
In regular expressions, the order of operation is closure (\*), concatenation ( $\bullet$ ) and then union (+).

For example, below are the regular expressions and their corresponding regular language.

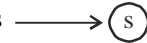
- $0^*10^* = \{w | w \text{ contains a single } 1\}$
- $1^*(01^*)^* = \{w | \text{every } 0 \text{ in } w \text{ is followed by at least one } 1\}$
- $\Sigma^*1\Sigma^* = \{w | w \text{ contains at least one } 1\}$

Any regular expression can be converted into a finite automaton that recognizes the language, it describes and vice versa. Some basic conversions from regular expression to finite automaton (M) are shown below:-

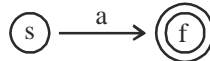
- If  $r = \epsilon$ , M is



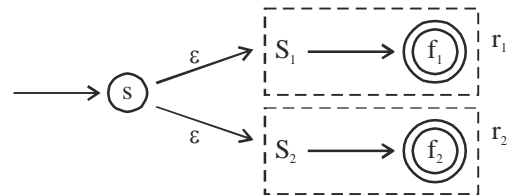
- If  $r = \phi$ , M is



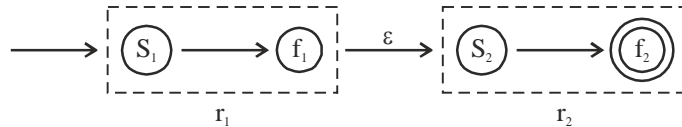
- If  $r = \{a\}$  for some  $a \in \Sigma$ , M is



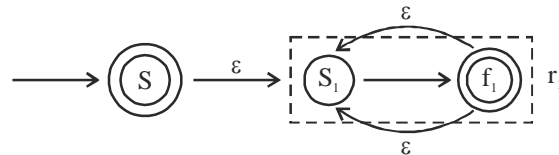
- If  $r = r_1 + r_2$ , then M is



- If  $r = r_1 \cdot r_2$ , then M is



- If  $r = r_1^*$ , then M is

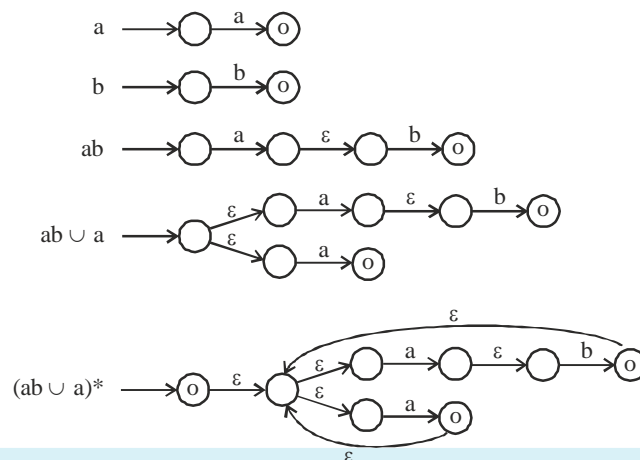


Using the above basic rules / examples, any regular expression can be converted into a finite automata.

Let's look at an example for better understanding

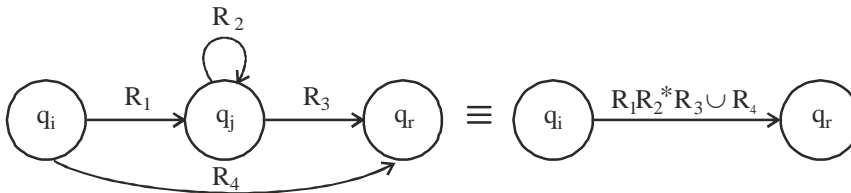
$$r = (ab \cup a)^*$$

To convert the regular expression  $(ab \cup a)^*$  into a finite automaton, we will first convert the regular expressions that are combined to get this one.

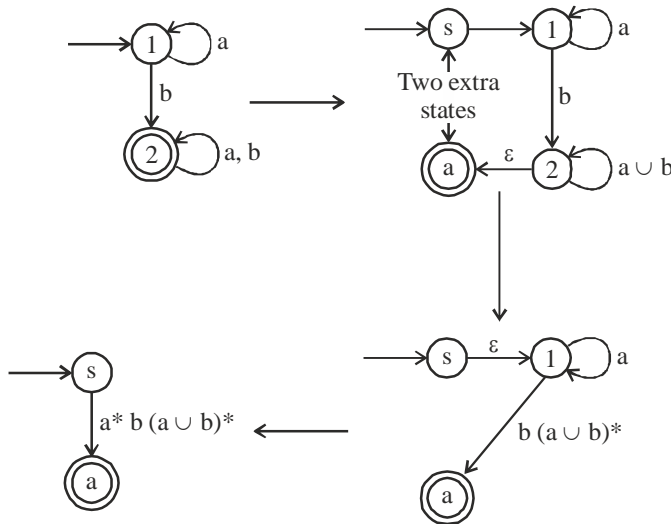


252

In this way, we can build NFA from a regular expression. Now let's convert a DFA into a regular language. For converting this, we have to follow a very simple procedure shown below:-



We have to remove states one by one to get only two states and therefore corresponding regular expression. Also, first we have to add two extra states, one start and one accept. Then we can start removing other states. For example,



### Properties (Identities) of regular expression

1.  $\phi + R = R$
2.  $\phi R + R\phi = \phi$
3.  $\wedge R = R \wedge = R$
4.  $\wedge * = \wedge$  and  $\phi * = \wedge$
5.  $R + R = R$
6.  $R * R * = R *$
7.  $RR * = R * R$
8.  $(R *) * = R *$
9.  $\wedge + RR * = \wedge + R * R = R *$
10.  $(PQ) * P = P (QP) *$
11.  $(P + Q) * = (P * Q *) = (P * + Q *) *$
12.  $(P + Q) R = PR + QR$  and  $R (P + Q) = RP + RQ$

### Arden's Theorem

Let P and Q be two regular expressions over  $\Sigma$ . If P does not contain  $\wedge$ , then the following equation in R, namely

$$R = Q + RP$$

has a unique solution, i.e.,  $R = QP^*$

equivalence of two regular expressions

Prove that  $(1 + 00^* 1) + (1 + 00^* 1) (0 + 10^* 1) * (0 + 10^* 1)$

is equal to  $0^* 1(0 + 10^* 1)^*$

$$\Rightarrow (1 + 00^* 1) + (1 + 00^* 1) (0 + 10^* 1)^* (0 + 10^* 1)$$

$$\Rightarrow (1 + 00^* 1) \wedge + (1 + 00^* 1) (0 + 10^* 1)^* (0 + 10^* 1)$$

$$\{R \cdot \wedge = R\}$$

$$\begin{aligned}
 &= (1 + 00^* 1) (\wedge + (0 + 10^* 1) (0 + 10^* 1)) \\
 &= (1 + 00^* 1) (0 + 10^* 1)^* \quad (\wedge RR^* = R^*) \\
 &= (\wedge. 1 + 00^* 1) (0 + 10^* 1)^* \\
 &= (\wedge + 00^*) \wedge (0 + 10^* 1)^* \\
 &= 0^* 1(0 + 10^* 1)^* \\
 &= \text{R.H.S.}
 \end{aligned}$$

### Regular expressions over an alphabet $\Sigma$

- each symbol  $a \in \Sigma$  is a regular expression
  - $\varepsilon$  is a regular expression
  - $\emptyset$  is a regular expression
  - if  $r$  and  $s$  are regular expressions, then so is  $(r|s)$
  - if  $r$  and  $s$  are regular expressions, then so is  $r|s$
  - if  $r$  is a regular expression, then so is  $(r)^*$
- Every regular expression is built up inductively, by *finitely many* applications of the above rules.

(N.B. we assume  $\varepsilon, \emptyset, (, ), |$ , and  $*$  are *not* symbols in  $\Sigma$ .)

We assume implicitly that the alphabet  $\Sigma$  does not contain the six symbols

$\varepsilon \quad \emptyset \quad ( \quad ) \quad | \quad *$

Then, concretely speaking, the regular expressions over  $\in$  form a certain set of strings over the alphabet obtained by adding these six symbols to  $\in$ . However it makes things more readable if we adopt a slightly more abstract syntax, dropping as many brackets as possible and using the convention that  $-^*$  binds more tightly than  $-$ ,  $-|$  binds more tightly than  $-$ .

So, for example,  $r|st^*$  means  $(r|s(t)^*)$ , not  $(r|s)(t)^*$ , or  $((r|st))^*$ , etc.

### Matching strings to regular expressions

- $u$  matches  $a \in \Sigma$  iff  $u = a$
- $u$  matches  $\varepsilon$  iff  $u = \varepsilon$
- no string matches  $\emptyset$
- $u$  matches  $r|s$  iff  $u$  matches either  $r$  or  $s$
- $u$  matches  $rs$  iff it can be expressed as the concatenation of two strings,  $u = vw$ , with  $v$  matching  $r$  and  $w$  matching  $s$
- $u$  matches  $r^*$  iff either  $u = \varepsilon$ , or  $u$  matches  $r$ , or  $u$  can be expressed as the concatenation of two or more strings, each of which matches  $r$

The definition of ‘ $u$  matches  $r^*$ ’, on Slide 6 is equivalent to saying

for some  $n \geq 0$ ,  $u$  can be expressed as a concatenation of  $n$  strings,  $u = u_1 u_2 \dots u_n$ , where each  $u_i$  matches  $r$ .

254

The case  $n = 0$  just means that  $u = \varepsilon$  (so  $\varepsilon$  always matches  $r^*$ ); and the case  $n = 1$  just means that  $u$  matches  $r$  (so any string matching  $r$  also matches  $r^*$ ). For example, if  $\Sigma = \{a, b, c\}$  and  $r = ab$ , then the strings matching  $r^*$  are

$\varepsilon, ab, abab, ababab, \text{ etc.}$

Note that we didn't include a regular expression for the '\*' occurring in the UNIX examples on Slide 1. However, *once we know which alphabet we are referring to*,  $\Sigma = \{a_1, a_2, \dots, a_n\}$  say, we can get the effect of \* using the regular expression

$$(a_1 a_2 \dots a_n)^*$$

Examples of matching, with $\Sigma = \{0, 1\}$
<ul style="list-style-type: none"> <li>• <math>0 1</math> is matched by each symbol in <math>\Sigma</math></li> <li>• <math>1(0 1)^*</math> is matched by any string in <math>\Sigma^*</math> that starts with a '1'</li> <li>• <math>((0 1)^*(0 1))^*</math> is matched by any string of even length in <math>\Sigma^*</math></li> <li>• <math>(0 1)^*(0 1)^*</math> is matched by any string in <math>\Sigma^*</math></li> <li>• <math>(\varepsilon 0)(\varepsilon 1) 11</math> is matched by just the strings <math>\varepsilon, 0, 1, 01</math>, and <math>11</math></li> <li>• <math>\emptyset 0</math> is just matched by <math>0</math></li> </ul>

### Examples based on Design of regular expression

**Example 6.** Design a regular expression for inputs  $a$  and  $b$   $\{a, b\}$  that contains exactly 2  $a$ 's.

**Sol.** Here, the description defines that there will be only 2  $a$ 's. In between number of  $b$ 's is not defined so  $b$  can be present in string or  $b$  may not be present in the string. This can be represent as  $b^*$  thus,

$$R = b^* ab^* ab^*$$

**Example 7.** Design a regular expression for inputs  $a$  and  $b$  that contains atleast  $2a$ 's.

**Sol.** Here at least  $2a$ 's should be there in string irrespective of position. For string length more than 2 the sequence of  $a, b$  is not defined. If the sequence and length of inputs  $a, b$  are not defined it is represented as

$$(a + b)^*$$

Thus regular will be written as

$$(a + b)^* a (a + b)^* b (a + b)^*$$

**Example 8.** Design a regular expression over strings  $a$  and  $b$  that contains at most 2  $a$ 's.

**Sol.** Here, the string can have either 1  $a$  or 2  $a$ 's or string will have no  $a$ 's.

For string containing 2  $a$ 's regular expression will be

$$b^* ab^* ab^*$$

For string containing 1  $a$ 's the regular expression will be  $b^* ab^*$  for string containing no  $a$ 's the regular expression will be.

Thus, the regular expression for language containing at most 2  $a$ 's will be

$$R = b^* ab^* + b^* ab^* ab^* + b^*$$

**Example 9.** Find the regular expression containing the sub-string  $aa$ .

**Sol.** The regular expression will be as

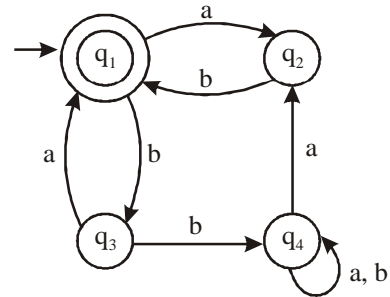
$$(a + b)^* aa(a + b)^*$$

### Rules for Regular Expressions

1.  $R \cup \phi = R$
2.  $R \circ \varepsilon = R$
3.  $\phi^* = \varepsilon$
4.  $\varepsilon^* = \varepsilon$



**Example 10.** Find the regular expression defined by following DFA



**Sol.** The equations for states will be

$$q_1 = \wedge + q_2b + q_3a$$

$$q_2 = q_1a$$

$$q_3 = q_1b$$

$$q_4 = q_3b + q_2a + q_4a + q_4b$$

For final state  $q_1$  equation is

$$q_1 = \wedge + q_2b + q_3a$$

$$= \wedge + q_1ab + q_1ba$$

(By putting value  $q_2 = q_1a$  and  $q_3 = q_1b$ )

$$= \wedge + q_1(ab + ba)$$

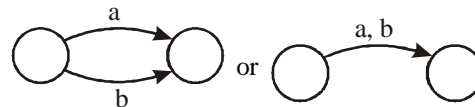
$$q_1 = \wedge (ab + ba)^*$$

$$(ab + ba)^*$$

(By Arden theorem  $Q = \wedge$ ,  $p = (ab + ba)$ ,  $R = Q + RP$  has solution  $R = QP^*$ )

### Construction of Finite Automata from Regular Expression

For regular expression  $(a + b)$  the FA will be



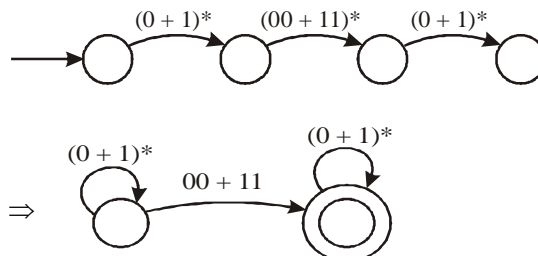
For regular expression  $ab$  the FA will be

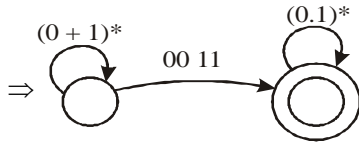


For regular expression  $a^*$  the FA will be

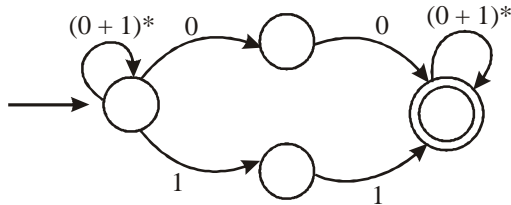


**Example 11.** Construct the finite automata for regular expression  $(0 + 1)^* (00 + 11) (0 + 1)^*$





Sol.



## PUMPING LEMMA

If  $A$  is a regular language, then there is a number  $P$  (the pumping length) where, if  $S$  is any string in  $A$  of length at least  $P$ , then  $S$  may be divided into three pieces,  $S = xyz$ , satisfying the following conditions:-

1. For each  $i \geq 0$ ,  $xy^iz \in A$
  2.  $|y| > 0$ , and
  3.  $|xy| \leq P$ .
- If a language does not satisfy these conditions, then it is a non-regular language.

### How to use the Pumping Lemma to prove that a language $L$ is not regular

For each  $\ell \geq 1$ , find some  $w \in L$  of length  $\geq \ell$  so that

$$(\dagger) \begin{cases} \text{no matter how } w \text{ is split into three, } w = u_1vu_2, \\ \text{with } \text{length}(u_1v) \leq \ell \text{ and } \text{length}(v) \geq 1. \\ \text{there is some } n \geq 0 \text{ for which } u_1v^nu_2 \text{ is not in } L. \end{cases}$$

Slide 31

### Examples

- (i)  $L_1 \stackrel{\text{def}}{=} \{a^n b^n \mid n \geq 0\}$  is not regular.  
[for each  $\ell \geq 1, a^\ell b^\ell \in L_1$  is of length  $\geq \ell$  and has property  $(\dagger)$  on Slide 31.]
- (ii)  $L_2 \stackrel{\text{def}}{=} \{w \in \{a, b\}^* \mid w \text{ a palindrome}\}$  is not regular.  
[For each  $\ell \geq 1, a^\ell b a^\ell \in L_1$  is of length  $\geq \ell$  and has property  $(\dagger)$ .]
- (iii)  $L_3 \stackrel{\text{def}}{=} \{a^p \mid p \text{ prime}\}$  is not regular.  
[For each  $\ell \geq 1$ , we can find a prime  $p$  with  $p > 2\ell$  and then  $a^p \in L_3$  has length  $\geq \ell$  and has property  $(\dagger)$ .]

## CONTEXT FREE LANGUAGES

The class of context free languages generalizes the class of regular languages, i.e. every regular language is a context free language. The reverse of this is not true. Informally, a context free language (CFL) is a language generated by a context free grammar (CFG). A CFG is a set of rules for deriving (or generating) strings (or sentences) in a language.

Informally, a CFG consists of:–

- A set of replacement rules, each having a LHS and a RHS.
- Two types of symbols, variables and terminals.
- LHS of each rule is a single variable (no terminals).
- RHS of each rule is string of zero or more variables and terminals.
- A string consists of only terminals.

Formally, a CFG is a 4-tuple  $(V, \Sigma, R, S)$ , where

1.  $V$  is a finite set called variables.
2.  $\Sigma$  is a finite set, disjoint from  $V$ , called terminals.
3.  $R$  is a finite set of rules, with each rule being a variable and a string of variables and terminals.
4.  $S \in V$  is the start variable.

For example,

$G = (\{S\}, \{a, b\}, R, S)$ , set of rules  $R$  is,

$S \rightarrow a S b \mid S S \mid \epsilon$ . This can be also written as

$$S \rightarrow a S b$$

$$S \rightarrow S S$$

$$S \rightarrow \epsilon$$

This grammar generates strings such as  $abab$ ,  $aaabbb$  and  $aababb$ .

- All the strings generated by the grammar constitute the language of the grammar and is called context free language (CFG).

### Chomsky Normal Form

A context free grammar is in Chomsky Normal Form if every rule is of the form.

$$A \rightarrow BC$$

$$A \rightarrow a$$

where,  $a$  is any terminal and  $A, B$  and  $C$  are any variables except  $B$  and  $C$  may not be the start variable. In addition, the rule  $S \rightarrow \epsilon$ , where  $S$  is the start material is allowed.

### Backus Naur Form

It is quite likely that the same non-terminal will appear on the left-hand side of several productions in a context-free grammar. Because of this, it is common to use a more compact notation for specifying productions, called *Backus-Naur Form* (BNF), in which all the productions for a given non-terminal are specified together, with the different right-hand sides being separated by the symbol ‘|’. BNF also tends to use the symbol ‘::=’ rather than ‘ $\rightarrow$ ’ in the notation for productions. An example of a context-free grammar in BNF is given

#### Example of Backus-Naur Form (BNF)

Terminals:

$$x \quad ' \quad + \quad - \quad * \quad ( \quad )$$

Non-terminals:

$$\text{id} \quad \text{op} \quad \text{exp}$$

Start symbol:

$$\text{exp}$$

Productions:

$$\text{id} ::= x \mid \text{id}'$$

$$\text{op} ::= + \mid - \mid *$$

$$\text{exp} ::= \text{id} \mid \text{exp op exp} \mid (\text{exp})$$

258

## Push Down Automata

Push Down Automata are like non-deterministic finite automata but have an extra component called a stack. The stack provides additional memory beyond the finite automata provides.

Formally, a push down automata is a 6-tuple  $(Q, \Sigma, \Gamma, \delta, q_0, F)$ , where

1.  $Q$  is the set of states.
2.  $\Sigma$  is the input alphabet.
3.  $\Gamma$  is the stack alphabet.
4.  $\delta : Q \times \Sigma_{\epsilon} \times \Gamma_{\epsilon} \rightarrow P(Q \times \Gamma_{\epsilon})$  is the transition function.
5.  $q_0 \in Q$  is the start state, and
6.  $F \subseteq Q$  is the set of accept states.

Below is the description of PDA that recognizes the language  $\{0^n 1^n | n \geq 0\}$ . Let  $M_1$  be  $(Q, \Sigma, \Gamma, \delta, q_1, F)$ , where

$Q = \{q_1, q_2, q_3, q_4\}$

$\Sigma = \{0, 1\}$

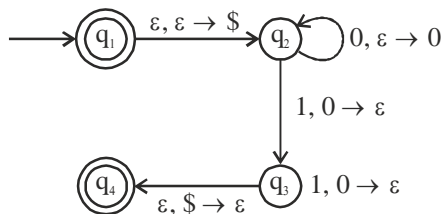
$\Gamma = \{0, \$\}$   $\$$  = This signifies that stack is empty

$F = \{q_1, q_4\}$  and

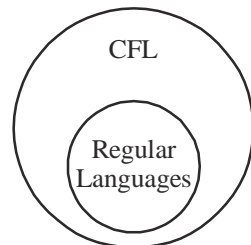
$\delta$  is given by the following table, where in blank entries signify  $\phi$ .

Input	0			1			$\varepsilon$		
Stock	0	\$	$\varepsilon$	0	\$	$\varepsilon$	0	\$	$\varepsilon$
$q_1$	$\{(q_2, \$)\}$								
$q_2$	$\{(q_2, 0)\}$			$\{(q_3, \varepsilon)\}$					
$q_3$				$\{(q_3, \varepsilon)\}$			$\{(q_4, \varepsilon)\}$		
$q_4$									

We can also use a state diagram to describe a PDA, as shown below:-



- A language is context free if and only if some push down automata recognizes it.
- Every regular language is context free language.



## Pumping Lemma for CFL

If  $A$  is a CFL, then there is a number  $P$  (the pumping length) where, if  $S$  is any string in  $A$  of length at least  $P$ , then  $S$  may be divided into five pieces  $S = uvwx$  satisfying the conditions:-

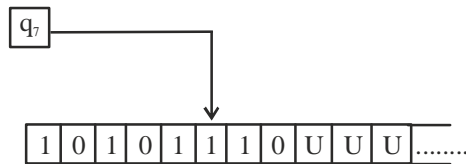
1. for each  $i \geq 0$ ,  $uv^iwx^iy \in A$ ,
2.  $|vx| > 0$ , and
3.  $|vwx| \leq P$ .

## TURING MACHINES

A turing machine is a 7-tuple  $(Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ , where

1.  $Q$  is the set of states.
2.  $\Sigma$  is the input alphabet not containing the blank symbol,  $\sqcup$ .
3.  $\Gamma$  is the tape alphabet, where  $\sqcup \in \Gamma$  and  $\Sigma \subseteq \Gamma$ .
4.  $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$  is the transition function.
5.  $q_0 \in Q$  is the start state.
6.  $q_{\text{accept}} \in Q$  is the accept state.
7.  $q_{\text{reject}} \in Q$  is the reject state, where  $q_{\text{reject}} \neq q_{\text{accept}}$ .

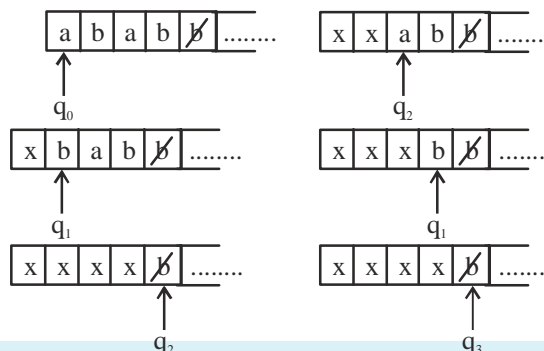
Initially  $M$  (turing machine) receives its input  $w \in \Sigma^*$  on the leftmost  $n$  squares of the tape and rest of the tape is blank, i.e. filled with blank symbol. The head starts from the leftmost square of the tape. Once  $M$  has started, the computation proceeds according to the rules described by the transition function. If  $M$  ever tries to move its head to the left off the left hand end of the tape, the head stays in the same place for that move, even though the transition function indicates  $L$ . The transition continues until it enters either the accept or reject states at which point it halts. If neither occurs,  $M$  goes on forever. As a TM computes, changes occur in the current state, current tape content and current head location. A setting of these three items is called a CONFIGURATION.



- The above diagram shows a TM with configuration  $10101q_7 110$ .  
Let  $C_1$  and  $C_2$  be configuration of  $M$ .  $C_1$  yields  $C_2$  if  $M$  is in configuration  $C_1$  after running  $M$  in configuration  $C_1$  for one step. Suppose  $\delta(q_1, b) = (q_2, C, L)$ , then  $aaq_1 bb$  yields  $aq_2 acb$ . Let  $w \in \Sigma^*$  and  $M$  be a turing machine.  $M$  accepts  $w$  if there are configs  $C_0, C_1, \dots, C_k$ , s.t.  
—  $C_0 = q_0 w$   
—  $C_i$  yields  $C_{i+1}$  for  $i = 0, \dots, k-1$ , and  
—  $C_k$  contains the accept state  $q_{\text{accept}}$ .
- A Turing Machine  $M$  recognizes a language  $L$  if  $M$  accepts exactly those strings in  $L$ .
- A language  $L$  is called recognizable or recursively enumerable if some TM recognizes  $L$ .
- A TM decides a language  $L$  if  $M$  accepts all strings in  $L$  and rejects all strings not in  $L$ .
- A language is called decidable or recursive if some TM decides  $L$ .
- $L$  is decidable if and only if both  $L$  and  $\neg L$  are recognizable.

**Example 12:** Construct a Turing machine which will accept the set of strings over  $\Sigma = \{a, b\}$  beginning with a 'a' and ending with a 'b'.

**Sol:**  $M = (K, \Sigma, \Gamma, \delta, q_0, F)$  where  
 $K = \{q_0, q_1, q_2, q_3\}; F = \{q_3\}$   
 $\Sigma = \{a, b\}; \Gamma = \{a, b, X, \text{blank}\}$   
 $\delta$  is defined as follows:-  
 $\delta(q_0, a) = (q_1, X, R)$   
 $\delta(q_1, a) = (q_1, X, R)$   
 $\delta(q_1, b) = (q_2, X, R)$   
 $\delta(q_2, a) = (q_1, X, R)$   
 $\delta(q_2, b) = (q_2, X, R)$   
 $\delta(q_2, \text{blank}) = (q_3, \text{halt})$



Let us see how the machine accepts abab.

It can be seen that after initially reading 'a', the machine goes to state  $q_1$ .

Afterwards, if it sees a 'a' it goes to state  $q_1$ ; if it sees 'b' it goes to  $q_2$ . Hence when it sees the leftmost blank symbol, if it is in state  $q_2$ , it accepts as this means that the last symbol read is a 'b'.

### Multitape Turing Machines

A multitape turing machine is like an ordinary turing machine with several tapes. The transition function is changed to allow for reading, writing and moving the heads on some or all of the tapes simultaneously.

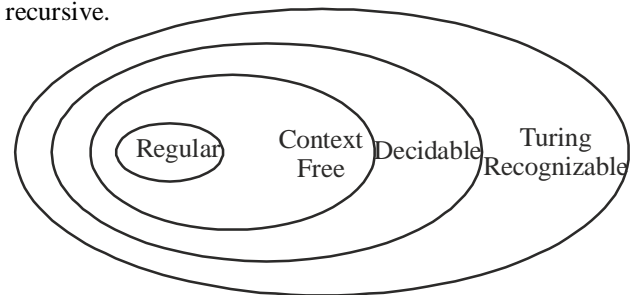
- Every multitape turing machine has an equivalent single-tape turing machine.

### Closure Properties of Recursive Languages

- If  $L$  is a recursive language over  $\Sigma$ , the  $\bar{L}$  is recursive.
- If  $L_1$  and  $L_2$  be recursive languages over  $\Sigma$ . Then  $L_1 \cup L_2$  is recursive.
- If  $L_1$  and  $L_2$  be recursive languages over  $\Sigma$ . The  $L_1 \cap L_2$  is recursive.

### Closure Properties of Recursively Enumerable Languages

- If  $L_1$  and  $L_2$  are recursively enumerable languages over  $\Sigma$ . Then  $L_1 \cup L_2$  is also recursively enumerable.
- If  $L_1$  and  $L_2$  are recursively enumerable languages over  $\Sigma$ . Then  $L_1 \cap L_2$  is also recursively enumerable.
- If  $L$  is a recursive language, then  $L$  is recursively enumerable.
- Relationship among classes of languages is shown below:–



### Halting Problem

- If a machine loops on input  $\langle M, w \rangle$ , i.e.  $M$  loops on  $w$ , then this machine does not decide the corresponding language. If the algorithm had some way to determine that  $M$  was not halting on  $w$ , it could reject. Hence, this type of problem is called halting problem. In other words, language is undecidable.
- A language is decidable if it is Turing-recognizable and co-Turing-recognizable. (We say this if it is the complement of a Turing-recognizable language).

# Past GATE Questions Exercise

1. Consider the following two problems on undirected graphs:  
 $\alpha$  — Given,  $G(V, E)$ , does  $G$  have an independent set of size  $|V| - 4$ ?

$\beta$  — Given,  $G(V, E)$ , does  $G$  have an independent set of size 5?  
 Which one of the following is true? [2006, 1 mark]

- (a)  $\alpha$  is in P and  $\beta$  is NP-complete
- (b)  $\alpha$  is NP-complete  $\beta$  is in P
- (c) Both  $\alpha$  and  $\beta$  are NP-complete
- (d) Both  $\alpha$  and  $\beta$  are in P

2. Consider the languages: [2006, 1 mark]

$$L_1 = \{WW^R \mid W \in \{0, 1\}^*\}$$

$$L_2 = \{W \# W^R \mid W \in \{0, 1\}^*\} \text{ where } \# \text{ is a special symbol}$$

$$L_3 = \{W W \mid W \in \{0, 1\}^*\}$$

Which one of the following is true?

- (a)  $L_1$  is a deterministic CFL
- (b)  $L_2$  is a deterministic CFL
- (c)  $L_3$  is a CFL but not a deterministic CFC
- (d)  $L_3$  is a deterministic CFL

3. Let  $L_1$  be a recursive language, and let  $L_2$  be a recursively enumerable but not a recursive language. Which one of the following is true? [2006, 1 mark]

- (a)  $\overline{L_1}$  is recursive and  $\overline{L_2}$  is recursively enumerable
- (b)  $\overline{L_1}$  is recursive and  $\overline{L_2}$  is not recursively enumerable
- (c)  $\overline{L_1}$  and  $\overline{L_2}$  are recursively enumerable
- (d)  $\overline{L_1}$  is recursively enumerable and  $\overline{L_2}$  is recursive

4. Consider the languages [2006, 1 mark]

$$L_1 = \{a^n b^n c^m \mid n, m > 0\} \text{ and } L_2 = \{a^n b^m c^m \mid n, m > 0\}$$

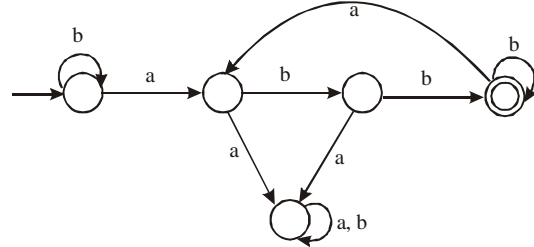
Which one of the following statements is false?

- (a)  $L_1 \cap L_2$  is a context-free language
- (b)  $L_1 \cup L_2$  is a context-free language
- (c)  $L_1$  and  $L_2$  are context-free language
- (d)  $L_1 \cap L_2$  is a context-sensitive language

5. Let  $N_f$  and  $N_p$  denote the classes of languages accepted by non-deterministic finite automata and non-deterministic push-down automata, respectively. Let  $D_f$  and  $D_p$  denote the classes of languages accepted by deterministic finite automata and deterministic push-down automata respectively. Which one of the following is true? [2006, 1 mark]

- (a)  $D_f \subset N_f$  and  $D_p \subset N_p$
- (b)  $D_f \subset N_f$  and  $D_p = N_p$
- (c)  $D_f = N_f$  and  $D_p = N_p$
- (d)  $D_f = N_f$  and  $D_p \subset N_p$

6. Consider the machine M: [2006, 1 mark]



The language recognized by M is

- (a)  $\{W \in (a, b)^* \mid \text{every } a \text{ in } W \text{ is followed by exactly two } b\text{'s}\}$
  - (b)  $\{W \in (a, b)^* \mid \text{every } a \text{ in } W \text{ is followed by at least two } b\text{'s}\}$
  - (c)  $\{W \in (a, b)^* \mid W \text{ contains the substring } abb\}$
  - (d)  $\{W \in (a, b)^* \mid W \text{ does not contain } aa \text{ as a substring}\}$
7. Consider three decision problems  $P_1$ ,  $P_2$  and  $P_3$ . It is known that  $P_1$  is decidable and  $P_2$  is undecidable. Which one of the following is true? [2006, 1 mark]
- (a)  $P_3$  is decidable if  $P_1$  is reducible to  $P_3$
  - (b)  $P_3$  is undecidable if  $P_3$  is reducible to  $P_2$
  - (c)  $P_3$  is undecidable if  $P_2$  is reducible to  $P_3$
  - (d)  $P_3$  is decidable if  $P_3$  is reducible to  $P_2$ 's complement
8. Consider the regular language  $L = (111 + 111111)^*$ . The minimum number of states in any DFA accepting this language is [2006, 1 mark]
- (a) 3
  - (b) 5
  - (c) 8
  - (d) 9
9. Let  $L_1$  be a regular language,  $L_2$  be a deterministic context-free language and  $L_3$  a recursively enumerable but not recursive language. Which one of the following statements is false? [2006, 1 mark]
- (a)  $L_1 \cap L_2$  is a deterministic CFL
  - (b)  $L_2 \cap L_1$  is recursive
  - (c)  $L_1 \cap L_2$  is context-free
  - (d)  $L_1 \cap L_2 \cap L_3$  is recursively enumerable

10. Consider the following statements about the context-free grammar: [2006, 1 mark]

$$G = \{S \rightarrow SS, S \rightarrow ab, S \rightarrow ba, S \rightarrow \epsilon\}$$

- 1.  $G$  is ambiguous.
- 2.  $G$  produces all strings with equal number of  $a$ 's and  $b$ 's.
- 3.  $G$  can be accepted by a deterministic PDA.

Which combination below expresses all the true statements about  $G$ ?

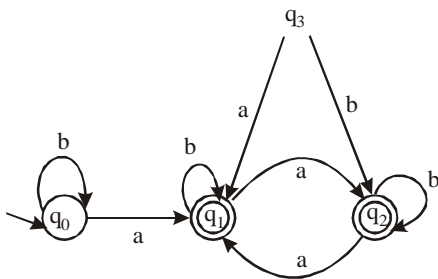
- (a) 1 only
- (b) 1 and 3
- (c) 2 and 3
- (d) 1, 2 and 3

262

11. Let  $SHAM_3$  be the problem of finding a Hamiltonian cycle in a graph  $G = (V, E)$  with  $|V|$  divisible by 3 and  $DHAM_3$  be the problem of determining if a Hamiltonian cycle exists in such graphs. Which one of the following is true? [2006, 1 mark]
- Both  $DHAM_3$  and  $SHAM_3$  are NP-hard
  - $SHAM_3$  are NP-hard but  $DHAM_3$  is not
  - $DHAM_3$  is NP-hard, and  $SHAM_3$  is not
  - Neither  $DHAM_3$  nor  $SHAM_3$  is NP-hard
12. For  $S \in (0+1)^*$  let  $d(s)$  denotes the decimal value of  $s$  (e.g.,  $d(101) = 5$ ) [2006, 1 mark]
- Which one of the following statements is true?
- $L$  is recursively enumerable but not recursive
  - $L$  is recursively but not context-free
  - $L$  is context-free but not regular
  - $L$  is regular
13. If  $s$  is a string over  $(0+1)^*$  then let  $n_0(s)$  denotes the number of 0's in  $s$  and  $n_1(s)$  the number of 1's in  $s$ . Which one of the following languages is not regular? [2006, 1 mark]
- $L = \{s \in (0+1)^* \mid n_0(s) \text{ is a 3-digit prime}\}$
  - $L = \{s \in (0+1)^* \mid \text{for every prefix } s' \text{ of } s, |n_0(s') - n_1(s')| \leq 2\}$
  - $L = \{s \in (0+1)^* \mid n_0(s) - n_1(s) \leq 4\}$
  - $L = \{s \in (0+1)^* \mid n_0(s) \bmod 7 = n_1(s) \bmod 5 = 0\}$
14. Let  $L_1 = \{0^{n+m}1^n0^m \mid n, m \geq 0\}$ ,  $L_2 = \{0^{n+m}1^{n+m}0^m \mid n, m \geq 0\}$ , and  $L_3 = \{0^{n+m}1^{n+m}0^{n+m} \mid n, m \geq 0\}$ . Which of these languages is/are not context free? [2006, 1 mark]
- $L_1$  only
  - $L_3$  only
  - $L_1$  and  $L_2$
  - $L_2$  and  $L_3$
15. Let  $S$  be an NP-complete problem and  $Q$  and  $R$  be two other problems not known to be in NP.  $Q$  is polynomial time reducible to  $S$  and  $S$  is polynomial time reducible to  $R$ . Which one of the following statements is true? [2006, 1 mark]
- $R$  is NP-complete
  - $R$  is NP-hard
  - $Q$  is NP-complete
  - $Q$  is NP-hard

**Common Data for Questions 16 and 17 :**

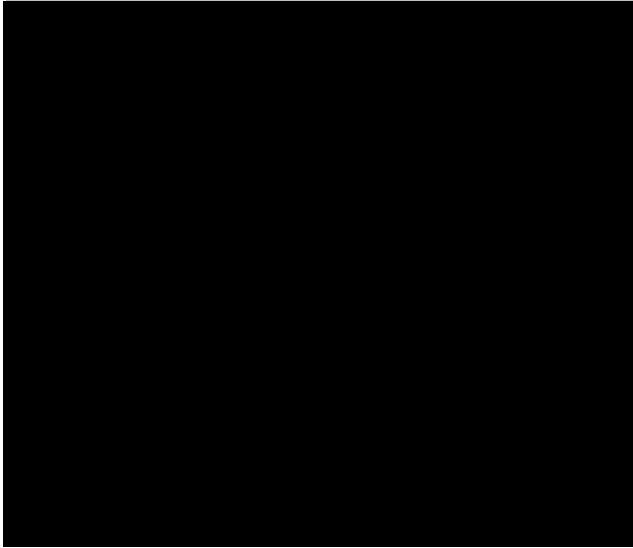
Consider the following Finite State Automation:



16. The language accepted by this automation is given by the regular expression [2007, 2 marks]
- $b^*ab^*ab^*ab$
  - $(a+b)^*$
  - $b^*a(a+b)^*$
  - $b^*ab^*ab^*$

17. The minimum state automation equivalent to the above FSA has the following number of states [2007, 2 marks]
- 1
  - 2
  - 3
  - 4
18. Which of the following languages is regular? [2007, 2 marks]
- $\{WW^R \mid W \in \{0,1\}^+\}$
  - $\{WW^RX \mid X, W \in \{0,1\}^+\}$
  - $\{WXW^R \mid X, W \in \{0,1\}^+\}$
  - $\{XWW^R \mid X, W \in \{0,1\}^+\}$
19. The language  $L = \{0^i21^i \mid i \geq 0\}$  over the alphabet  $\{0, 1, 2\}$  is [2007, 2 marks]
- not recursive
  - is recursive and is a deterministic CFL
  - is a regular language
  - is not a deterministic CFL but a CFL
20. A minimum state deterministic finite automaton accepting the language  $L = \{W \mid W \in \{0, 1\}^*, \text{ number of 0's and 1's in } W \text{ are divisible by 3 and 5, respectively}\}$ . [2007, 2 marks]
- 15 states
  - 11 states
  - 10 states
  - 9 states
21. Which of the following is true? [2007, 1 mark]
- Every subset of a regular set is regular
  - Every finite subset of a non-regular set is regular
  - The union of two non-regular sets is regular
  - Infinite union of finite sets is regular
22. Which of the following problems is undecidable? [2007, 1 mark]
- Membership problem for CFGs
  - Ambiguity problem for CFGs
  - Finiteness problem for FSAs
  - Equivalent problem for FSAs
23. Which of the following are regular sets? [2008, 2 marks]
- $\{a^n b^{2m} \mid n \geq 0, m \geq 0\}$
  - $\{a^n b^m \mid n = 2m\}$
  - $\{a^n b^m \mid n \neq m\}$
  - $\{xyc \mid x, y, c \in \{a, b\}^*\}$
- 1 and 4
  - 1 and 3
  - 1 only
  - 4 only
24. Match the List I with List II and select the correct answer using the codes given below the lists. [2008, 2 marks]





- (a) E - P, F - R, G - Q, H - S  
(b) E - R, F - P, G - S, H - Q  
(c) E - R, F - P, G - Q, H - S  
(d) E - P, F - R, G - S, H - Q

25. Which of the following statements are true? [2008, 2 marks]

- Every left-recursive grammar can be converted to a right-recursive grammar and vice-versa.
- All  $\epsilon$ -productions can be removed from any context-free grammar by suitable transformations.
- The language generated by a context-free grammar all of whose productions are of the form  $X \rightarrow W$  or  $X \rightarrow WY$  (where,  $W$  is a string of terminals and  $Y$  is non-terminal), is always regular.
- The derivation trees of strings generated by a context-free grammar in Chomsky Normal Form are always binary trees.

- (a) 1, 2, 3 and 4 (b) 2, 3 and 4  
(c) 1, 3 and 4 (d) 1, 2 and 4

26. Given below are two finite state automata ( $\rightarrow$  indicates the start state and F indicates a final state) [2008, 2 marks]

Y:

	a	b
$\rightarrow 1$	1	2
2 (F)	2	1

Z:

	a	b
$\rightarrow 1$	2	2
2 (F)	1	1

Which of the following represents the product automaton  $Z \times Y$ ?

(a)

	a	b
$\rightarrow P$	Q	R
Q	R	S
R(F)	Q	P
S	Q	P

(b)

	a	b
$\rightarrow P$	S	Q
Q	R	S
R(F)	Q	P
S	P	Q

(c)

	a	b
$\rightarrow P$	Q	S
Q	R	S
R(F)	Q	P
S	Q	P

(d)

	a	b
$\rightarrow P$	S	Q
Q	S	R
R(F)	Q	P
S	Q	P

27. Which of the following statements is false? [2008, 2 marks]

- (a) Every NFA can be converted to an equivalent DFA  
(b) Every non-deterministic turing machine can be converted to an equivalent deterministic turing machine  
(c) Every regular language is also a context-free language  
(d) Every subset of a recursively enumerable set is recursive

28. If  $L$  and  $\bar{L}$  are recursively enumerable, then  $L$  is

[2008, 1 mark]

- (a) regular (b) context-free  
(c) context-sensitive (d) recursive

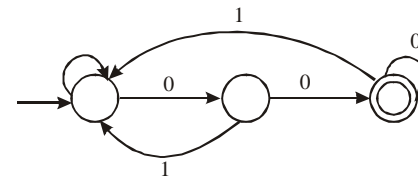
29. Which of the following are decidable? [2008, 1 mark]

- Whether the intersection of two regular languages is infinite.
  - Whether a given context-free language is regular.
  - Whether two push-down automata accept the same language.
  - Whether a given grammar is context-free
- (a) 1 and 2 (b) 1 and 4  
(c) 2 and 3 (d) 2 and 4

30. Which of the following is true for the language  $\{a^p \mid p \text{ is a prime}\}$ ? [2008, 1 mark]

- (a) It is not accepted by a turning machine  
(b) It is regular but not context-free  
(c) It is context-free but not regular  
(d) It is neither regular nor context-free, but accepted by a turing machine

31.



The above DFA accepts the set of all strings over  $\{0, 1\}$  that

- (a) begin either with 0 or 1 [2009, 2 marks]  
(b) end with 0  
(c) end with 00  
(d) contain the substring 00

32. Let  $L_1 = L_1 \cap L_2$ , where  $L_1$  and  $L_2$  are languages as defined below

$$L_1 = \{a^m b^m c a^n b^n \mid m, n \geq 0\}$$

$$L_2 = \{a^i b^j c^k \mid i, j, k \geq 0\}$$

Then  $L$  is

[2009, 2 marks]

264

- (a) not recursive  
(b) regular  
(c) context-free but not regular  
(d) recursively enumerable but not context-free

33. Given the following state table of an FSM with two states A and B, one input and one output

Present State A	Present State B	Input	Next State A	Next State B	Output
0	0	0	0	0	1
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	1	0	0
0	0	1	0	1	0
0	1	1	0	0	1
1	0	1	0	1	1
1	1	1	0	0	1

If the initial state is A = 0, B = 0, what is the minimum length of an input string, which will take the machine to the state A = 0, B = 1 with output = 1? [2009, 2 marks]

- (a) 3 (b) 4  
(c) 5 (d) 6

34. Match all items in List I with correct options from those given in List II [2009, 1 mark]

List I	List II
P. Regular expression	1. Syntax analysis
Q. Push-down 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

35. Which one of the following is false? [2009, 1 mark]

- (a) There is 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

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

- (a) The set of all strings containing the substring 00  
(b) The set of all strings containing at most 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.

37.  $S \rightarrow aSa \mid bSb \mid a \mid b$ ; The language generated by the above grammar over the alphabet {a, b} is the set of [2009, 1 mark]

- (a) all palindromes  
(b) all odd length palindromes  
(c) strings that begin and with the same symbol  
(d) all even length palindromes

38. Let W be any string of length n in {0, 1}\*. Let L be the set of all sub-strings of W. What is the minimum number of states in a non-deterministic finite automaton that accept L? [2010, 2 marks]

- (a) n - 1 (b) n  
(c) n + 1 (d)  $2^{n-1}$

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

- (a) Only  $L_2$  is context-free  
(b)  $L_2$  and  $L_3$  are context-free  
(c)  $L_1$  and  $L_2$  is context-free  
(d) All are context free

40. Let  $L = \{W \in (0, 1)^* \mid W \text{ has even number of 1s}\}$ , 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, 2 marks]

- (a)  $(0^* 10^* 1)^*$  (b)  $0^* (10^* 10^*)^*$   
(c)  $0^* (10^* 1)^* 0^*$  (d)  $0^* 1(10^* 1)^* 10^*$

41. Let  $L_1$  be a recursive language. Let  $L_2$  and  $L_3$  be languages that are recursively enumerable but not recursive. Which of the following statements is not necessarily true? [2010, 1 mark]

- (a)  $L_2 - L_1$  is recursively enumerable  
(b)  $L_1 - L_3$  is recursively enumerable  
(c)  $L_2 \cap L_1$  is recursively enumerable  
(d)  $L_2 \cup L_1$  is recursively enumerable

42. Consider the languages,  $L_1$ ,  $L_2$  and  $L_3$  as given below

$$L_1 = \{0^p 1^q \mid p, q \in \mathbb{N}\} \quad [2011, 2 \text{ marks}]$$

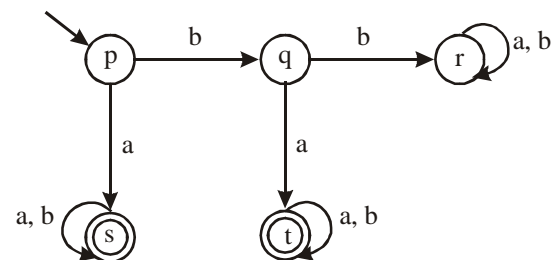
$$L_2 = \{0^p 1^q \mid p, q \in \mathbb{N} \text{ and } p = q\} \text{ and}$$

$$L_3 = \{0^p 1^q 0^r \mid p, q, r \in \mathbb{N} \text{ and } p = q = r\}$$

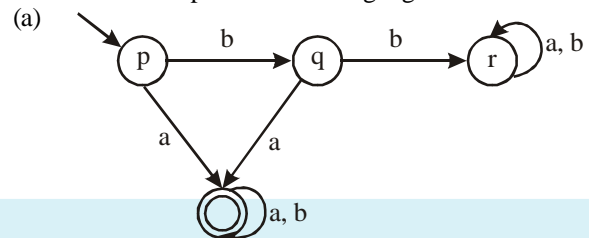
Which of the following statements is not true?

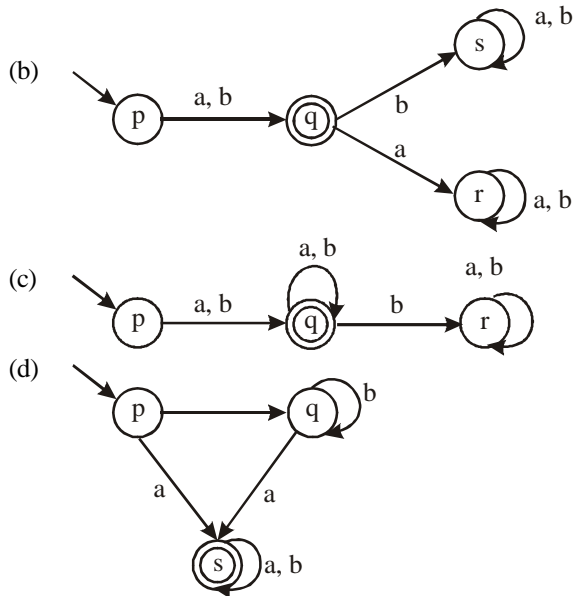
- (a) Push Down Automata (PDA) can be used to recognize  $L_1$  and  $L_2$   
(b)  $L_1$  is a regular language.  
(c) All the three languages are context-free  
(d) Turning machines can be used to recognize all the languages

43. A deterministic finite automaton (DFA) D with alphabet  $\Sigma = \{a, b\}$  is given below. [2011, 2 marks]



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





44. Definition of a language  $L$  with alphabet  $\{a\}$  is given as following and  $n$  is a positive integer constant. What is the minimum number of states needed in a DFA to recognize  $L$ ? [2011, 2 marks]

- (a)  $k+1$  (b)  $n+1$   
(c)  $2n+1$  (d)  $2k+1$

45. Which of the following pairs have different expressive power? [2011, 1 mark]

- (a) Deterministic Finite Automata (DFA) and Non-deterministic Finite Automata (NFA)  
(b) Deterministic Push Down Automata (DPDA) and Non-deterministic Push Down Automata (NPDA)  
(c) Deterministic single-tape turing machine and non-deterministic single tape turing machine  
(d) Single-tape turing machine and multi-tape turing machine

46. 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  $\{Q^n p^n q^n \mid n \in \mathbb{N}\}$ . Then which of the following is always regular? [2011, 1 mark]

- (a)  $P \cap Q$  (b)  $P - Q$   
(c)  $\Sigma^* - P$  (d)  $\Sigma^* - Q$

47. What is the minimal form of the Karnaugh map shown below? Assume that  $X$  denotes a don't care term. [2012, 2 marks]

ab \ cd	00	01	11	10
00	1	X	X	1
01	X			1
11				
10	1			X

- (a)  $\overline{bd}$  (b)  $\overline{bd} + \overline{bc}$   
(c)  $\overline{bd} + a\overline{bcd}$  (d)  $\overline{bd} + \overline{bc} + \overline{cd}$

48. A list of  $n$  strings, each of length  $n$ , is sorted into lexicographic order using the merge-sort algorithm. The worst case running time of this computation is [2012, 2 marks]

- (a)  $O(n \log n)$  (b)  $O(n^2 \log n)$   
(c)  $O(n^2 + \log n)$  (d)  $O(n^2)$

49. The worst case running time to search for an element in a balanced binary search tree with  $n^{2^n}$  elements is [2012, 1 mark]

- (a)  $\Theta(n \log n)$  (b)  $\Theta(n^{2^n})$   
(c)  $\Theta(n)$  (d)  $\Theta(\log n)$

50. The recurrence relation capturing the optimal execution time of the Towers of Hanoi problem with  $n$  discs is [2012, 1 mark]

- (a)  $T(n) = 2T(n-2) + 2$  (b)  $T(n) = 2T(n-1) + n$   
(c)  $T(n) = 2T(n/2) + 1$  (d)  $T(n) = 2T(n-1) + 1$

51. Let  $W(n)$  and  $A(n)$  denote respectively, the worst case and average case running time of an algorithm executed on an input of size  $n$ . Which of the following is always true? [2012, 1 mark]

- (a)  $A(n) = \Omega(W(n))$  (b)  $A(n) = \Theta(W(n))$   
(c)  $A(n) = O(W(n))$  (d)  $A(n) = o(W(n))$

52. Consider the languages  $L_1 = \Phi$  and  $L_2 = \{a\}$ . Which one of the following represents  $L_1 L_2^* \cup L_1^*$ ? [2013, 1 Mark]

- (a)  $\{\epsilon\}$  (b)  $\Phi$   
(c)  $a^*$  (d)  $(\epsilon, a)$

53. Which of the following statements is/are false?

- For every non-deterministic Turing machine, there exists an equivalent deterministic Turing machine.
- Turing recognisable languages are closed under union and complementation.
- Turing decidable languages are closed under intersection and complementation.
- Turing recognisable languages are closed under union and intersection.

- [2013, 1 Mark]  
(a) 1 and 4 only (b) 1 and 3 only  
(c) 2 only (d) 3 only

54. Which of the following is/are undecidable?

- $G$  is a CFG. Is  $L(G) = \Phi$ ?
- $G$  is a CFG. Is  $L(G) = \Sigma^*$ ?
- $M$  is a Turing machine. Is  $L(M)$  regular?
- $A$  is a DFA and  $N$  is an NFA. Is  $L(A) = L(N)$ ?

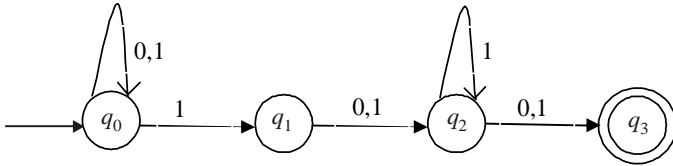
- [2013, 2 Marks]  
(a) 3 only (b) 3 and 4 only  
(c) 1, 2 and 3 only (d) 2 and 3 only

55. Which one of the following is TRUE?

- [2014, Set-1, 1 Mark]  
(a) The language  $L = \{a^n b^n \mid n \geq 0\}$  is regular.  
(b) The language  $L = \{a^n \mid n \text{ is prime}\}$  is regular.  
(c) The language  $L = \{w \mid w \text{ has } 3k+1 \text{ b's for some } k \in \mathbb{N} \text{ with } \Sigma = \{a, b\}\}$  is regular.  
(d) The language  $L = \{ww \mid w \in \Sigma^* \text{ with } \Sigma = \{0, 1\}\}$  is regular.

266

56. Consider the finite automaton in the following figure.



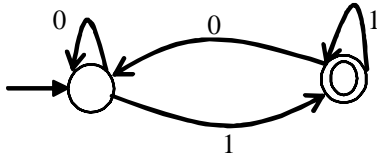
Which is the set of reachable states for the input string 0011?  
[2014, Set-1, 1 Mark]

- (a)  $\{q_0, q_1, q_2\}$  (b)  $\{q_0, q_1\}$   
(c)  $\{q_0, q_1, q_2, q_3\}$  (d)  $\{q_3\}$

57. Let  $L$  be a language and  $\bar{L}$  be its complement. Which one of the following is NOT a viable possibility?  
[2014, Set-1, 2 Marks]

- (a) Neither  $L$  nor  $\bar{L}$  is recursively enumerable (r.e.).  
(b) One of  $L$  and  $\bar{L}$  is r.e. but not recursive; the other is not r.e.  
(c) Both  $L$  and  $\bar{L}$  are r.e. but not recursive.  
(d) Both  $L$  and  $\bar{L}$  are recursive.

58. Which of the regular expressions given below represent the following DFA?  
[2014, Set-1, 2 Marks]



- I  $0^*1(1+00^*1)^*$   
II  $0^*1^*1+11^*0^*1$   
III  $(0+1)^*1$

- (a) I and II only (b) I and III only  
(c) II and III only (d) I, II, and III

59. If  $L_1 = \{a^n \mid n \geq 0\}$  and  $L_2 = \{b^n \mid n \geq 0\}$ , consider

[2014, Set-2, 1 Mark]

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

Which one of the following is CORRECT?

- (a) Only (I) (b) Only (II)  
(c) Both (I) and (II) (d) Neither (I) nor (II)

60. Let  $A \leq_m B$  denotes that language  $A$  is mapping reducible (also known as many-to-one reducible) to language  $B$ . Which one of the following is FALSE?

[2014, Set-2, 1 Mark]

- (a) If  $A \leq_m B$  and  $B$  is recursive then  $A$  is recursive.  
(b) If  $A \leq_m B$  and  $A$  is undecidable then  $B$  is undecidable.  
(c) If  $A \leq_m B$  and  $B$  is recursively enumerable then  $A$  is recursively enumerable.  
(d) If  $A \leq_m B$  and  $B$  is not recursively enumerable then  $A$  is not recursively enumerable.

61. Let  $\langle M \rangle$  be the encoding of a Turing machine as a string over  $\Sigma = \{0, 1\}$ . Let  $L = \{\langle M \rangle \mid M \text{ is a Turing machine that accepts a string of length } 2014\}$ . Then,  $L$  is

[2014, Set-2, 2 Marks]

- (a) decidable and recursively enumerable  
(b) undecidable but recursively enumerable  
(c) undecidable and not recursively enumerable  
(d) decidable but not recursively enumerable

62. Let  $L_1 = \{w \in \{0, 1\}^* \mid w \text{ has at least as many occurrences of } (110)\text{'s as } (011)\text{'s}\}$ . Let  $L_2 = \{w \in \{0, 1\}^* \mid w \text{ has at least as many occurrences of } (000)\text{'s as } (111)\text{'s}\}$ . Which one of the following is TRUE?  
[2014, Set-2, 2 Marks]

- (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

63. The length of the shortest string NOT in the language (over  $\Sigma = \{a, b\}$ ) of the following regular expression is \_\_\_\_\_.  
[2014, Set-3, 1 Mark]

$a^*b^*(ba)^*a^*$

64. Let  $\Sigma$  be a finite non-empty alphabet and let  $2^{\Sigma^*}$  be the power set of  $\Sigma^*$ . Which one of the following is TRUE?

[2014, Set-3, 1 Mark]

- (a) Both  $2^{\Sigma^*}$  and  $\Sigma^*$  are countable  
(b)  $2^{\Sigma^*}$  is countable and  $\Sigma^*$  is uncountable  
(c)  $2^{\Sigma^*}$  is uncountable and  $\Sigma^*$  is countable  
(d) Both  $2^{\Sigma^*}$  and  $\Sigma^*$  are uncountable

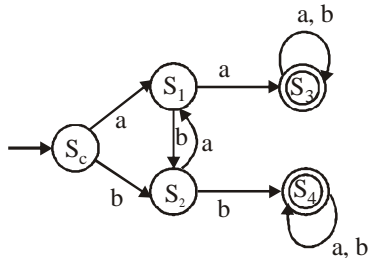
# Practice Exercise

- Context-free languages and regular languages are both closed under the operations (s) of
  - Union
  - Intersection
  - Concatenation
  - (i) and (ii) only
  - (ii) and (iii) only
  - (i) and (iii) only
  - all of the above
- Let  $r_1 = ab^*c^*$  &  $r_2 = (a^*b \vee c)^*$  and  $r_3 = (a \vee b \vee c)^*$ . Then which of the following is true
  - $w = ac$  belongs to  $L(r_2)$  and  $L(r_3)$  but not  $L(r_1)$
  - $w = ac$  belongs to  $L(r_3)$  only
  - $w = ac$  belongs to  $L(r_1)$ ,  $L(r_2)$  and  $L(r_3)$
  - $w = ac$  belongs to  $L(r_1)$  and  $L(r_3)$  but not  $L(r_2)$
- Let  $\Sigma = \{a, b\}$   
 $r_1 = a(a \vee b)^*$   
 $r_2 = b(a \vee b)^*$   
 Which of the following is true?
  - $L(r_1) = L(r_2) = \Sigma^*$
  - $L(r_1) \cap L(r_2) = \{\lambda\}$
  - $L(r_1) \cup L(r_2) = \Sigma^*$
  - $L(r_1) \cup L(r_2) \cup \{\lambda\} = \Sigma^*$
- Which of the following statements are true?
  - $abcd \in L((b^*a)^*(d^*c)^*)$
  - $abcd \in L((d^*c^*b^*a)^*)$
  - $abcd \in L((a^*b^*c^*d)^*)$
  - (i) and (iii) only
  - (ii) and (iii) only
  - (i) and (ii) only
  - all of the above
- Which of the following regular expression corresponds to the language of all strings over the alphabet  $\{a, b\}$  that do not end with  $ab$ .
  - $(a + b)^*(aa + ba + bb)$
  - $(a + b)^*(aa + ba + bb) + a + b + \lambda$
  - $b^*ab^*a$
  - $b^*aab^*$
- How many minimum number of states are required in the DFA (over the alphabet  $\{a, b\}$ ) accepting all the strings with the number of  $a$ 's divisible by 4 and number of  $b$ 's divisible by 5?
  - 20
  - 9
  - 7
  - 15
- How many states does the DFA constructed for the set of all strings ending with "00", have?
  - 2
  - 3
  - 4
  - 5
- Let  $X = \{0, 1\}$ ,  $L = X^*$  and  $R = \{0^n 1^n / n > 0\}$  then the language  $L \cup R$  and  $R$  respectively
  - Regular, Regular
  - Non regular, Regular
  - Regular, Not regular
  - Not regular, Not Regular
- Consider  $L_1 = 0^n 1^n$ ,  $L_2 = 0^n c 1^n$   
 Which of the following statements are correct?
  - $L_1$  and  $L_2$  are accepted by non-deterministic PDA.
  - $L_1$  and  $L_2$  are accepted by deterministic PDA.
  - Only  $L_2$  is accepted by deterministic PDA.
  - only (i)
  - (i) and (ii)
  - (i) and (iii)
  - All three
- Let  $L_1$  and  $L_2$  are regular sets defined over the alphabet  $\Sigma^*$ . Mark the false statement
  - $L_1 \cup L_2$  is regular
  - $L_1 \cap L_2$  is not regular
  - $\Sigma^* - L_1$  is regular
  - $L_1^*$  is regular
- Which of the following functions are computable with turning Machine?
  - $n^*(n-1)^*(n-2)^* \dots * 2^* 1^*$
  - $\lceil \log_2 n \rceil$
  - $2^{2^n}$
  - all of the above
- Let  $L_1 = \{a^n b^n c^n, n \geq 0\}$   
 $L_2 = \{a^{2n} b^{2n} c^{2n}, n \geq 0\}$   
 $L_3 = \{a^{2n} b^{2n} c^n, n \geq 0\}$ 
  - $L_1 \subseteq L_2$  and  $L_3 \subseteq L_2$
  - $L_2 \subseteq L_1$  and  $L_2 \subseteq L_3$
  - $L_2 \subseteq L_1$  but  $L_2 \not\subseteq L_3$
  - $L_1 \subseteq L_2$  and  $L_2 \subseteq L_3$
- Which of the following regular expression does not represent strings beginning with atleast one 0 and ends with at least one 1?
  - $0^* 1^*$
  - $00^*(0+1)^* 1$
  - $0(0+1)^* 1$
  - None of the above
- The following CFG  
 $S \rightarrow aS \mid bS \mid a \mid b$   
 is equivalent to the regular expression
  - $(a^* + b^*)$
  - $(a + b)^+$
  - $(a + b)(a + b)^*$
  - $(a + b)^*(a + b)$
  - 2 and 3 only
  - 2, 3 and 4
  - All of the above
  - 3 and 4 only
- The Moore machine has six tuples  $(Q, \Sigma, \Delta, \delta, \lambda, d_0)$ . Which of the following is true?
  - $\delta$  is the output function
  - $\delta$  is the transition function  $\Sigma$  into  $Q$
  - $\lambda$  is the transition function  $\Sigma \times Q$  into  $Q$
  - $\lambda$  is the output function mapping  $Q$  into  $\Delta$ .
- Suppose  $r_1 = \epsilon$ ,  $r_2 = 0^* 1^*$ , which of the following statement is true about  $r_1$  and  $r_2$ ?
  - $r_1$  is not regular expression, while  $r_2$  is a regular
  - $r_1$  and  $r_2$  both are regular expression
  - $r_1$  is regular expression, but  $r_2$  is not
  - Neither  $r_1$  nor  $r_2$  are regular expressions
- If  $L_1$  is regular and  $L_2$  is CFL over  $\Sigma^*$  which of the following statement is incorrect?
  - $L_1 \cup L_2$  is CFL
  - $L_1 \cap L_2$  is regular
  - $L_1^*$  is regular
  - None of the above



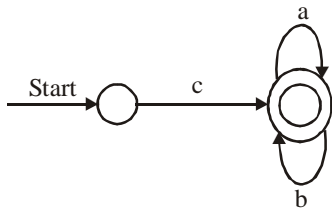
268

18.  $\Sigma^*$  over  $\{0, 1\}$  and  ${}_2\Sigma^*$  are respectively.  
 (a) Uncountably infinite and uncountably infinite  
 (b) Countably infinite and uncountably infinite  
 (c) Countably infinite and countably infinite  
 (d) None of the above
19. Which of the following languages is/are context free?  
 1.  $\{a^n b^n c^m d^m \mid n \geq 1, m \geq 1\}$   
 2.  $\{a^n b^m c^m d^n \mid n \geq 1, m \geq 1\}$   
 3.  $\{a^n b^m c^n d^m \mid n \geq 1, m \geq 1\}$   
 4.  $\{a^m b^n c^m d^n \mid n \geq 1, m \geq 1\}$   
 (a) 1 and 2 (b) 3 and 4  
 (c) 2 and 4 (d) 1, 2, 3 and 4
20. Consider the machine M shown below



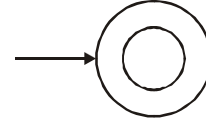
$L(M) = ?$

- (a)  $L(M) = \{\text{words starting with aa or bb}\}$   
 (b)  $L(M) = \{\text{words ending with aa or bb}\}$   
 (c)  $L(M) = \{\text{words containing aa or bb as a subword}\}$   
 (d) None of these
21. The regular expression for "Binary numbers that are multiples of two" is  
 (a)  $(0|1)^*1$  (b)  $(0|1)^*0$   
 (c)  $(1|0)^*1$  (d)  $(1|0)^*00$
22. Which of the string is accepted by given NDFA?

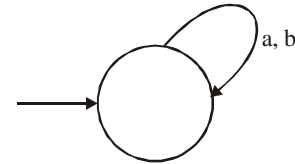


- (a)  $c.(a \cup b)^*$  (b)  $c.a^*b^*$   
 (c)  $c.(ab)^*$  (d) None of these
23. Consider the language  
 $L_1 = \{a^n b^m c^n d^m \mid n \geq 1, m \geq 1\}$  and  
 $L_2 = \{a^n b^m c^m d^n \mid n \geq 1, m \geq 1\}$   
 (a) Both  $L_1$  and  $L_2$  are context free  
 (b)  $L_1$  is not context free but  $L_2$  is context free  
 (c) both are not context free  
 (d)  $L_1$  is context free but  $L_2$  is not context free
24. The following GFG  
 $S \rightarrow aS \mid bS \mid a \mid b$  and  $S \rightarrow aS \mid bS \mid a \mid b \mid \lambda$  is equivalent to regular expressions  
 (a)  $(a+b)$  and  $\lambda + a + b$  respectively  
 (b)  $(a+b)$  and  $(a+b)^*$  respectively  
 (c)  $(a+b)$  and  $(\lambda + a + b)(\lambda + a + b)$  respectively  
 (d) None of these

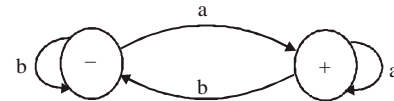
25. The statement, "ATM can't solve halting problem" is  
 (a) True (b) False  
 (c) Still a open question (d) None of these
26. Any given transition diagram has an equivalent  
 (a) regular expression (b) NDFSM  
 (c) DFSM (d) all of these
27. The FSM shown in the figure accepts



- (a) all strings (b) no strings  
 (c)  $\epsilon$ -alone (d) none of these
28. The FSM shown in the figure accepts

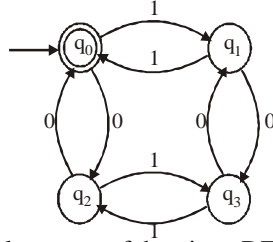


- (a) all strings (b) no strings  
 (c)  $\epsilon$ -alone (d) none of these
29. CFLs are not closed under  
 (a) Union (b) Concatenation  
 (c) Closure (d) Intersection
30. The set  $A = \{a^n b^n a^n \mid n = 1, 2, 3, \dots\}$  is an example of a language that is  
 (a) regular (b) not context-free  
 (c) context-free (d) none of these
31. Consider the two regular languages  $L_1 = (a+b)^*a$  and  $L_2 = b(a+b)^*$ . The intersection of  $L_1$  and  $L_2$  is given by  
 (a)  $(a+b)^*ab$  (b)  $ab(a+b)^*$   
 (c)  $a(a+b)^*b$  (d)  $b(a+b)^*a$
32. Which of the following languages over  $\{a, b, c\}$  is accepted by deterministic push down automata?  
 (a)  $\{\omega \mid \omega \text{ is palindrome over } \{a, b, c\}\}$   
 (b)  $\{\omega \omega^R \mid \omega \in \{a, b, c\}^*\}$   
 (c)  $\{a^n b^n c^n \mid n \geq 0\}$   
 (d)  $\{\omega C \omega^R \mid \omega \in \{a, b\}^*\}$
33. Consider the FA shown in the figure given below, where " $\rightarrow$ " is the start and " $+$ " is the ending state.  
 The language accepted by the FA is



- (a)  $(a+b)^*b$  (b)  $(a+b)^*a$   
 (c)  $a^*b$  (d)  $a^*b^*$
34. Regarding the power of recognizing the languages, which of the following statements is false?  
 (a) The NDFA are equivalent to DFA  
 (b) NDPDA are equivalent to DPDA  
 (c) NDTMs are equivalent to DTMs  
 (d) Multiple tape TMs are equivalent to single tape TMs
35. If  $\forall w \in \Sigma^*$  it can determined in finite time, whether or not  $w \in L$ , the L is  
 (a) decidable (b) undecidable  
 (c) non-deterministic (d) intractable

36. Consider the transition diagram of DFA as given below.



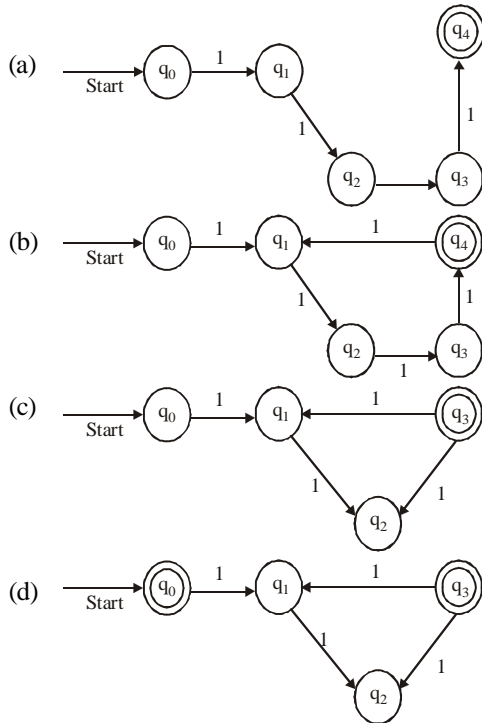
Which is the language of the given DFA?

- (a)  $L = \{\epsilon\}$   
 (b)  $L = \{\}$   
 (c)  $L = \{w | w \text{ has equal no. of 1's and 0's}\}$   
 (d) none of these

37. Let  $\Sigma = (0, 1)$ , then an automation A accepting only those words from  $\Sigma$  having an odd number of 1's requires \_\_\_\_\_ states including the start state.

- (a) 2 (b) 3  
 (c) 4 (d) 5

38. Design an FSM to check whether a given unary number is divisible by 3.



39. The given transition table is for the FSM that accepts a string if it ends with 'aa'. Which is the final states?

$\delta$	a	b
start	q <sub>0</sub>	q <sub>2</sub>
q <sub>0</sub>	q <sub>1</sub>	q <sub>2</sub>
q <sub>1</sub>	q <sub>1</sub>	q <sub>2</sub>
q <sub>2</sub>	q <sub>0</sub>	q <sub>2</sub>

- (a) q<sub>0</sub> (b) q<sub>1</sub>  
 (c) q<sub>2</sub> (d) can't be determined

40. A minimum state DFA accepting the same language has how many states?

- (a) 1 (b) 2  
 (c) 3 (d) 4

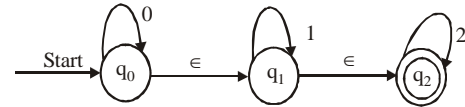
41.  $(a + b)^* a (a + b)^*$

- (a) strings of odd length having a at the middle  
 (b) all strings containing a and b  
 (c) all strings of odd length  
 (d) all strings containing atleast one 'a'

42. Let L be the set of all strings over  $\{0, 1\}$  of length 6 or less. Write a simple RE corresponding to L.

- (a)  $(a + 1)^*$  (b)  $(0 + 1)^6$   
 (c)  $(0 + 1 + \epsilon)^*$  (d)  $(0 + 1 + \epsilon)^6$

43. Given the R.E. described by the following NFA.



- (a)  $(012)^*$  (b)  $(0 + 1 + 2)^*$   
 (c)  $0^*1^*2^*$  (d) None of these

44.  $[(a + b)(a + b)]^*$

- (a) all strings  
 (b) all strings of even length  
 (c) all strings in which group of 2 symbols has both the symbols same but grouping must be done left to right starting from first symbol  
 (d) all strings in which group of 2 symbols has both the symbols same but grouping must be done right to left starting from last symbol.

45. If a DFA is represented by the following transition table, then how many states does the corresponding minimal DFA contains?

State	Input	
	0	1
(start) $\rightarrow$ A	B	C
B	B	D
C	B	C
D	B	E
(final) $\rightarrow$ E	B	C

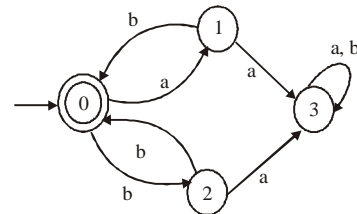
- (a) 2 (b) 3  
 (c) 4 (d) 5

46.  $G = \{(a, b), \{S\}, S, \{S \rightarrow b/Sa/aS/SS\})$

Which of the following are true?

- (i) aabbaa  $\in G$   
 (ii) G is ambiguous  
 (iii) regular expression corresponding to G is  $ba^*$   
 (a) (i) and (ii) only (b) (ii) and (iii) only  
 (c) (i) and (iii) only (d) all of the above

47. Which languages does the following DFA accept?



- (a)  $(ab)^*$  (b)  $(ab + bb)^*$   
 (c)  $(ab + ba)^*$  (d)  $(aa + bb)^*$

48. Which of the following statements are false?

- (a) The regular expression  $(1 + 10)^*$  denotes all strings of 0's and 1's beginning with '1' and not having two consecutive 0's

270

- (b) The regular expression  $(0+1)^*011$  denotes all strings of 0's and 1's ending with 011.
- (c) The regular expression  $OO(1+10)^*$  denotes all strings of 0's and 1's atleast two consecutive 0's
- (d) The regular expression  $(O+\epsilon)(1+10)^*$  denotes all strings of 0's and 1's that do not have two consecutive 0's
49. Which of the following languages can't be accepted by a deterministic PDA?
- (a) The set of palindromes over alphabet  $\{a, b\}$
- (b) The set of strings of balanced parenthesis
- (c)  $L = \{WcW^R \mid W \text{ in } (0+1)^*\}$
- (d)  $L = \{0^n 1^n \mid n \geq 0\}$
50. Which language does the following PDA accept  
 $M = \{\{q_0, q_1\}, \{0, 1\}, \{z_0, x\}, \delta, q_0, z_0, \phi\}$  and  $\delta$  is given by  
 $\delta(q_0, 0, z_0) = (q_0, xz_0)$   
 $\delta(q_0, 0, x) = (q_0, xx)$   
 $\delta(q_0, 1, x) = (q_1, x)$   
 $\delta(q_1, 1, x) = (q_1, \epsilon)$   
 $\delta(q_1, \epsilon, z_0) = (q_0, \epsilon)$
- (a)  $L = \{0^n 1^n \mid n \geq 0\}$  (b)  $L = \{0^n 1^n \mid n \geq 1\}$
- (c)  $L = \{0^n 1^{n+1} \mid n \geq 0\}$  (d)  $L = \{0^n 1^{n+1} \mid n \geq 1\}$
51. Which of the following statement must always be true for A and B? Suppose A and B are two sets of strings from  $\Sigma^*$ , such that  $B \subseteq A$
- (i) if A is finite then, B is finite
- (ii) if A is regular then, B is regular
- (iii) if A is context free then, B is context free
- (a) (i) only (b) (ii) only
- (c) (iii) only (d) All three
52. Consider the following grammar G  
 $S \rightarrow aA, A \rightarrow (aA, bB), B \rightarrow (bB, c, cC), C \rightarrow (c, cC)$   
 Which of the following is  $L(G)$ ?
- (a)  $L(G) = a^*b^*c^*$  (b)  $L(G) = aa^*b^*c^*$
- (c)  $L(G) = aa^*bb^*cc^*$  (d)  $L(G) = (abc)^*$
53. Consider the following set of languages  
 $L_1 = a^m b^n \mid n = m^2$   
 $L_2 = a^m b^m c^m d^n \mid m, n, > 0$   
 $L_3 = a^m b^m c^n d^n \mid m, n > 0$   
 Which of the above language is not context free?
- (a)  $L_1$  and  $L_3$  (b)  $L_2$  and  $L_3$
- (c)  $L_1$  and  $L_2$  (d) All  $L_1, L_2$  and  $L_3$
54.  $L_1$  has the following grammar  
 $S \rightarrow aB/BA$   
 $A \rightarrow bAA/aS/a$   
 $B \rightarrow bS/aBB$   
 $L_2$  has the following grammar  
 $S \rightarrow Sba/a$   
 Which of the following statement is true about?
- $L_3 = L_1 \cap L_2$  and  $L_4 = L_1 L_1^*$
- (a) Both  $L_3$  and  $L_4$  are not context free
- (b)  $L_3$  is context free but  $L_4$  is not
- (c) Both  $L_3$  and  $L_4$  are context free
- (d)  $L_4$  is context free, but not  $L_3$
55. Let M be a turing machine has  $Q = \{q_0, q_1, q_2, q_3, q_4\}$  a set of states, input alphabets  $\{0, 1\}$ . The tape alphabets are  $\{0, 1, B, X, Y\}$ . The symbol B is used to represent end of input string. The final state is  $q_4$ . The transistions are as follows.

- $(q_0, 0) = (q_1, X, R)$
- $(q_0, Y) = (q_3, Y, R)$
- $(q_1, 0) = (q_1, 0, R)$
- $(q_1, 1) = (q_2, Y, L)$
- $(q_1, Y) = (q_1, Y, R)$
- $(q_2, 0) = (q_2, 0, L)$
- $(q_2, X) = (q_0, X, R)$
- $(q_2, Y) = (q_2, Y, L)$
- $(q_3, Y) = (q_3, Y, R)$
- $(q_3, B) = (q_4, B, R)$

Which of the following is true about M?

- (a) M halts on L having 100 as substring
- (b) M halts on L having 101 as substring
- (c) M halts on  $= 0^n 1^n, n \geq 0$
- (d) M halts on L not having 1100 substring
56. Consider the following regular grammar,

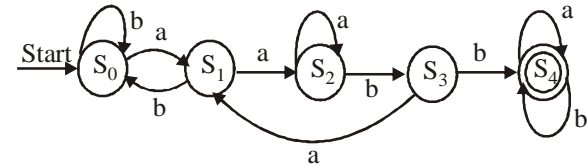
$$R_1 = (a|b)^*$$

$$R_2 = (a^* | b^*)^*$$

$$R_3 = (\epsilon a | b^*)^*$$

Minimized deterministic finite automata of which  $R_1, R_2$  and  $R_3$  are exactly same except state names?

- (a) DFA for  $R_1$  and  $R_2$  are similar
- (b) DFA for  $R_2$  and  $R_3$  are similar
- (c) DFAS for  $R_1, R_2$  and  $R_3$  are different
- (d) DFAS for  $R_1, R_2$  and  $R_3$  are similar
57. Consider the following machine M



Which is the language  $L(M)$  accepted by this machine?

- (a)  $L(M) = \{\text{Set of all words starting with aabb}\}$
- (b)  $L(M) = \{\text{Set of all words having aabb as a subword}\}$
- (c)  $L(M) = \{\text{Set of all words ending with aabb}\}$
- (d)  $L(M) = \{\text{Set of all words with exactly one occurrence of aabb}\}$
58. If  $L_1$  and  $L_2$  are a pair of complementary languages. Which of the following statement is not possible?
- (a) Both  $L_1$  and  $L_2$  are recursive
- (b)  $L_1$  is recursive and  $L_2$  is recursively enumerable but not a recursive
- (c) Neither  $L_1$  nor  $L_2$  recursively enumerable
- (d) One is recursively enumerable but not recursive, the other is not recursively enumerable.
59. Consider the grammar consisting of 7 productions  
 $S \rightarrow aA | aBB$   
 $A \rightarrow aaA | \lambda$   
 $B \rightarrow bB | bbC$   
 $C \rightarrow B$   
 After elimination of Unit, useless and  $\lambda$ -productions, how many production remain in the resulting grammar?
- (a) 2 (b) 3
- (c) 4 (d) 5
60. Referring to Turing machine in previous question, what will be the output when the input is  $I_1 = 0100$  and  $I_2 = 0010$ ?
- (a)  $I_1 = 0$  and  $I_2 = \text{Blank}$  (b)  $I_1 = \text{Blank}$  and  $I_2 = 0$
- (c)  $I_1 = 1$  and  $I_2 = 1$  (d) None of the above



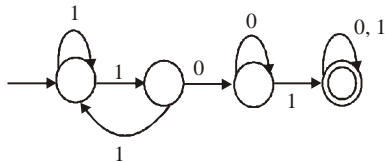
61. Suppose  $L_1$  and  $L_2$  are two language over  $\Sigma^*$

$$L = \Sigma^* - ((\Sigma^* - L_1) \cup (\Sigma^* - L_2))$$

$L_1$  and  $L_2$  are CFL.

Which of the following statement is true?

1.  $L$  is necessarily CFL
  2.  $L$  may or may not be CFL
  3.  $L_1 \subseteq L_2$
- (a) only 2 (b) 1 and 3  
(c) 2 and 3 (d) All are correct
62. In string of length  $n$ , how many proper prefixes can be generated
- (a)  $2^n$  (b)  $n$   
(c)  $n(n+1)/2$  (d)  $n-1$
63. The following CFG
- $$S \rightarrow aB \mid bA$$
- $$A \rightarrow ba \mid aS \mid bAA$$
- $$B \rightarrow b \mid bS \mid aBB$$
- generates strings of terminals that have
- (a) equal number of a's and b's
  - (b) odd number of a's and even number of b's
  - (c) even number of a's and even number of b's
  - (d) odd number of a's and even number of a's
64. The regular expression  $0^*(10^*)^*$  denotes the same set as
- (a)  $(1^*0)^*1^*$  (b)  $0 + (0+10)^*$   
(c)  $(0+1)^*10(0+1)^*$  (d) None of the above
65. Consider the following deterministic finite state automation M.



Let  $S$  denote the set of seven bit binary strings in which the first, the forth and the last bits are 1. The number of strings in  $S$  that are accepted by  $M$  is

- (a) 1 (b) 5  
(c) 7 (d) 8
66. The diagonalization language,  $L_d$  is a
- (a) recursive language
  - (b) recursive enumerable but not recursive
  - (c) non-recursivity - enumerable (non-RE) language
  - (d) both (b) and (c)
67. Time taken by one tape TM to simulate  $n$  moves of  $k$ -tape. TM is
- (a)  $O(n)$  (b)  $O(n^k)$   
(c)  $O(n^2)$  (d) none of these
68. Which of the following is true?
- (a) PDA with 2 stacks is equivalent to TM
  - (b) Regular expression can represents only some of the languages.
  - (c) Both (a) and (b)
  - (d) neither (a) nor (b)
69.  $L \in NP$  does not obviously imply that
- (a)  $L' \in NP$  (b) both (a) and (c)  
(c)  $L' \in P$  (d) none of these

70.  $P, Q, R$  are three languages. If  $P$  and  $R$  are regular and if  $PQ = R$ , then
- (a)  $Q$  has to be regular
  - (b)  $Q$  can not be regular
  - (c)  $Q$  need not be regular
  - (d)  $Q$  has to be a CFL
71. The Travelling Salesman Problem (TSP) is
- (a) NP but not NP complete
  - (b) NP complete
  - (c) Neither NP nor NP-complete
  - (d) None of these
72. Which of the following is false?
- (a) PATH is a P class problem
  - (b) Dijkstra's algorithm is a problem in P
  - (c) CLIQUE is a NP class problem
  - (d) RELPRIME is a NP class problem
73. If there is an NP-complete language  $L$  whose complement is in NP, then the complement of any language in NP is in
- (a) NP
  - (b) P
  - (c) Both (a) and (b)
  - (d) None of these
74. What is the regular expression for the language generated by  $S \rightarrow aS \mid bA, A \rightarrow d \mid cA$
- (a)  $a^*bd$
  - (b)  $a^*(bd)(bcc)^*d$
  - (c)  $a^*b(cc)^*d$
  - (d) None of these
75. Consider the grammar  $S \rightarrow PQ \mid SQ \mid PS, P \rightarrow x, Q \rightarrow y$ . To get string of  $n$  terminals, the number of productions to be used is
- (a)  $n^2$  (b)  $2n$   
(c)  $2^n + 1$  (d)  $2n - 1$
76. Which of the following can be recognized by a DFA?
- (a) The number of 1, 2, 4, ---  $2^n$  ---written in binary
  - (b) The number of 1, 2, 4, ---  $2^n$  ---written in unary
  - (c) The set of binary strings in which the number of 0's is same as the number of 1's
  - (d) The set  $\{0, 101, 11011, 1110111, \dots\}$
77. Give a RE for the set of strings which are either all b's or else there is an 'a' followed by some b's also containing  $\epsilon$ .
- (a)  $b^* + ab^*$  (b)  $(\epsilon + a)b^+$   
(c)  $b^* + ab^* + \epsilon$  (d)  $(\epsilon + a)b^+ + \epsilon$
78. Given an arbitrary non-deterministic finite automaton (NFA) with  $N$  states, the maximum number of states in an equivalent minimized DFA is at least.
- (a)  $N^2$  (b)  $2^N$   
(c)  $2N$  (d)  $N!$
79. Consider a DFA over  $\Sigma = \{a, b\}$  accepting all strings which have number of  $a$ 's divisible by 6 and number of  $b$ 's divisible by 8. What is the minimum number of states that the DFA will have?
- (a) 8 (b) 14  
(c) 15 (d) 48
80. Consider the following languages :
- $$L1 = \{ww \mid w \in \{a, b\}^*\}$$
- $$L2 = \{ww^R \mid w \in \{a, b\}^*, w^R \text{ is the reverse of } w\}$$
- $$L3 = \{0^{2i} \mid i \text{ is an integer}\}$$
- $$L4 = \{0^{i^2} \mid i \text{ is an integer}\}$$
- Which of the languages are regular?
- (a) Only  $L1$  and  $L2$  (b) Only  $L2, L3$  and  $L4$   
(c) Only  $L3$  and  $L4$  (d) Only  $L3$

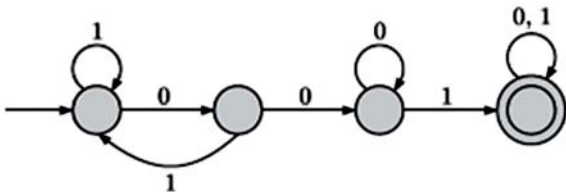
272

81. Consider the following problem  $x$ .  
 Given a Turing machine  $M$  over the input alphabet  $\Sigma$ , any state  $q$  of  $M$ .  
 And a word  $w \in \Sigma^*$  does the computation of  $M$  on  $w$  visit the state  $q$ ?  
 Which of the following statements about  $x$  is correct?  
 (a)  $x$  is decidable  
 (b)  $x$  is undecidable but partially decidable  
 (c)  $x$  is undecidable and not even partially decidable  
 (d)  $x$  is not a decision problem
82. The language accepted by a Pushdown Automaton in which the stack is limited to 10 items is best described as  
 (a) Context free (b) Regular  
 (c) Deterministic Context free (d) Recursive
83. Ram and Shyam have been asked to show that a certain problem  $\Pi$  is NP-complete. Ram shows a polynomial time reduction from the 3-SAT problem to  $\Pi$ , and Shyam shows a polynomial time reduction from  $\Pi$  to 3-SAT. Which of the following can be inferred from these reduction?  
 (a)  $\Pi$  is NP-hard but not NP-complete  
 (b)  $\Pi$  is in NP, but is not NP-complete  
 (c)  $\Pi$  is NP-complete  
 (d)  $\Pi$  is neither NP-hard, nor in NP
84. Nobody knows yet if  $P = NP$ . Consider the language  $L$  defined as follows

$$L = \begin{cases} (0+1)^* & \text{if } P = NP \\ \emptyset & \text{otherwise} \end{cases}$$

Which of the following statements is true?

- (a)  $L$  is recursive  
 (b)  $L$  is recursively enumerable but not recursive  
 (c)  $L$  is not recursively enumerable  
 (d) Whether  $L$  is recursive or not will be known after we find out if  $P = NP$
85. If the strings of a language  $L$  can be effectively enumerated in lexicographic (i.e., alphabetic) order, which of the following statements is true?  
 (a)  $L$  is necessarily finite  
 (b)  $L$  is regular but not necessarily finite  
 (c)  $L$  is context free but not necessarily regular  
 (d)  $L$  is recursive but not necessarily context free
86. Consider the following deterministic finite state automaton  $M$ .



Let  $S$  denote the set of seven bit binary strings in which the first, the fourth, and the last bits are 1. The number of strings in  $S$  that are accepted by  $M$  is

- (a) 1 (b) 5  
 (c) 7 (d) 8
87. Let  $G = (\{S\}, \{a, b\}, R, S)$  be a context free grammar where the rule set  $R$  is  
 $S \rightarrow a S b \mid S S \mid \epsilon$   
 Which of the following statements is true?

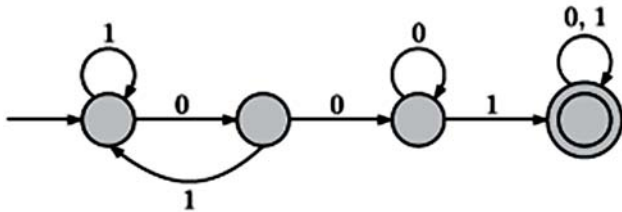
- (a)  $G$  is not ambiguous  
 (b) There exist  $x, y, \in L(G)$  such that  $xy \notin L(G)$   
 (c) There is a deterministic pushdown automaton that accepts  $L(G)$   
 (d) We can find a deterministic finite state automaton that accepts  $L(G)$
88. Consider the following two statements :  
 $S_1 : \{0^{2n} \mid n \geq 1\}$  is a regular language  
 $S_2 : \{0^m 1^n 0^{m+n} \mid m \geq 1 \text{ and } n \geq 1\}$  is a regular language  
 Which of the following statements is incorrect?  
 (a) Only  $S_1$  is correct  
 (b) Only  $S_2$  is correct  
 (c) Both  $S_1$  and  $S_2$  are correct  
 (d) None of  $S_1$  and  $S_2$  is correct
89. Which of the following statements true?  
 (a) If a language is context free it can be always be accepted by a deterministic push-down automaton.  
 (b) The union of two context free language is context free.  
 (c) The intersection of two context free language is context free  
 (d) The complement of a context free language is context free
90. The smallest finite automaton which accepts the language  $\{x/ \text{length of } x \text{ is divisible by } 3\}$  has  
 (a) 2 states (b) 3 states  
 (c) 4 states (d) 5 states
91. The C language is :  
 (a) A context free language  
 (b) A context sensitive language  
 (c) A regular language  
 (d) Parsable fully only by a Turing machine
92. The language accepted by a Pushdown Automaton in which the stack is limited to 10 items is best described as  
 (a) Context free (b) Regular  
 (c) Deterministic Context free (d) Recursive
93. Ram and Shyam have been asked to show that a certain problem  $\Pi$  is NP-complete. Ram shows a polynomial time reduction from the 3-SAT problem to  $\Pi$ , and Shyam shows a polynomial time reduction from  $\Pi$  to 3-SAT. Which of the following can be inferred from these reduction?  
 (a)  $\Pi$  is a NP-hard but not NP-complete  
 (b)  $\Pi$  is in NP, but is not NP-complete  
 (c)  $\Pi$  is NP-complete  
 (d)  $\Pi$  is neither NP-hard, nor in NP
94. Nobody knows yet if  $P = NP$ . Consider the language  $L$  defined as follows

$$L = \begin{cases} (0+1)^* & \text{if } P = NP \\ \emptyset & \text{otherwise} \end{cases}$$

Which of the following statements is true?

- (a)  $L$  is recursive  
 (b)  $L$  is recursively enumerable but not recursive  
 (c)  $L$  is not recursively enumerable  
 (d) Whether  $L$  is recursive or not will be known after we find out if  $P = NP$

95. If the strings of a language  $L$  can be effectively enumerated in lexicographic (i.e. alphabetic) order, which of the following statements is true?
- $L$  is necessarily finite
  - $L$  is regular but not necessarily finite
  - $L$  is context free but not necessarily regular
  - $L$  is recursive but not necessarily context free
96. Consider the following deterministic finite state automaton  $M$ .



Let  $S$  denote the set of seven bit binary strings in which the first, the fourth, and the last bits are 1. The number of strings in  $S$  that are accepted by  $M$  is

- 1
  - 5
  - 7
  - 8
97. Let  $G = (\{S\}, \{a, b\}, R, S)$  be a context free grammar where the rule set  $R$  is
- $$S \rightarrow a S b \mid S S \mid \epsilon$$
- Which of the following statements is true?
- $G$  is not ambiguous
  - There exist  $x, y, \in L(G)$  such that  $xy \notin L(G)$
  - There is a deterministic pushdown automaton that accepts  $L(G)$
  - We can find a deterministic finite state automaton that accepts  $L(G)$
98. A single tape Turing Machine  $M$  has two states  $q^0$  and  $q^1$ , of which  $q^0$  is the starting state. The tape alphabet of  $M$  is  $\{0, 1, B\}$  and its input alphabet is  $\{0, 1\}$ . The symbol  $B$  is the blank symbol used to indicate end of an input string. The transition function of  $M$  is described in the following table

	0	1	B
$q^0$	$q^{1,1,R}$	$q^{1,1,R}$	Halt
$q^1$	$q^{1,1,R}$	$q^{0,1,L}$	$qH0, B, L$

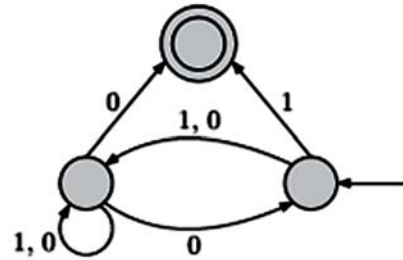
The table is interpreted as illustrated below.

The entry  $(q^{1,1,R})$  in row  $q^0$  and column 1 signifies that if  $M$  is in state  $q^0$  and reads 1 on the current tape square, then it writes 1 on the same tape square, moves its tape head one position to the right and transitions to state  $q^1$ .

Which of the following statements is true about  $M$ ?

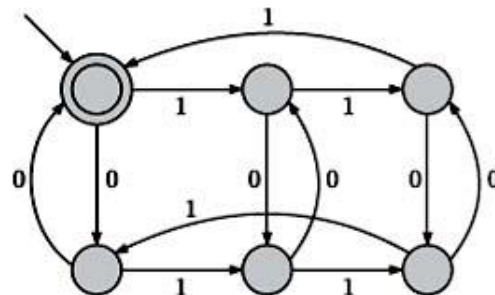
- $M$  does not halt on any string in  $(0+1)^+$
- $M$  does not halt on any string in  $(00+1)^*$
- $M$  halts on all string ending in a 0
- $M$  halts on all string ending in a 1

99. Consider the NFAM shown below.



Let the language accepted by  $M$  be  $L$ . Let  $L_1$  be the language accepted by the NFAM  $M_1$ , obtained by changing the accepting state of  $M$  to a non-accepting state and by changing the non-accepting state of  $M$  to accepting states. Which of the following statements is true?

- $L_1 = \{0, 1\}^* - L$
  - $L_1 = \{0, 1\}^*$
  - $L_1 \subseteq L$
  - $L_1 = L$
100. The problems 3-SAT and 2-SAT are
- both in P
  - both NP-complete
  - NP-complete and in P respectively
  - undecidable and NP-complete respectively
101. The following finite state machine accepts all those binary strings in which the number of 1's and 0's are respectively



- divisible by 3 and 2
  - odd and even
  - even and odd
  - divisible by 2 and 3
102. The language  $\{a^m b^{m+n} \mid m, n \leq 1\}$  is
- regular
  - context-free but not regular
  - context sensitive but not context free
  - type-0 but not context sensitive
103. Consider the following grammar  $C$

$$S \rightarrow bS \mid aA \mid b$$

$$A \rightarrow bA \mid aB$$

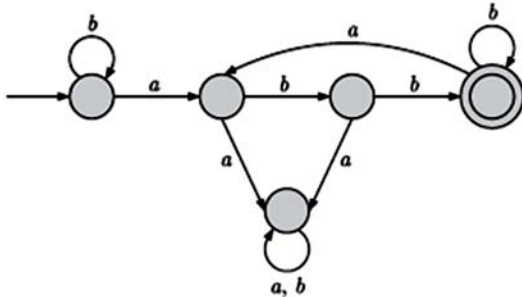
$$B \rightarrow bB \mid aS \mid a$$

Let  $N_a(W)$  and  $N_b(W)$  denote the number of  $a$ 's and  $b$ 's in a string  $W$  respectively. The language  $L(G) \subseteq \{a, b\}^+$  generated by  $G$  is

- $\{W \mid N_a(W) > 3N_b(W)\}$
- $\{W \mid N_b(W) > 3N_a(W)\}$
- $\{W \mid N_a(W) = 3k, k \in \{0, 1, 2, \dots\}\}$
- $\{W \mid N_b(W) = 3k, k \in \{0, 1, 2, \dots\}\}$

274

104. Consider three decision problem  $P_1, P_2$  and  $P_3$ . It is known that  $P_1$  is decidable and  $P_2$  is undecidable. Which one of the following is TRUE?
- $P_3$  is decidable if  $P_1$  is reducible to  $P_3$
  - $P_3$  is undecidable if  $P_3$  is reducible to  $P_2$
  - $P_3$  is undecidable if  $P_2$  is reducible to  $P_3$
  - $P_3$  is decidable if  $P_3$  is reducible to  $P_2$ 's complement
105. Consider the machine  $M$



The language recognised by  $M$  is

- $\{W \in \{a, b\}^* / \text{every } a \text{ in } w \text{ is followed by exactly two } b\text{'s}\}$
  - $\{W \in \{a, b\}^* / \text{every } a \text{ in } w \text{ is followed by at least two } b\text{'s}\}$
  - $\{W \in \{a, b\}^* / w \text{ contains the substring 'abb'}\}$
  - $\{W \in \{a, b\}^* / w \text{ does not contain 'aa' as a substring}\}$
106. Let  $L_1$  be a recursive language, and let  $L_2$  be a recursively enumerable but not a recursive language. Which one of the following is TRUE?
- $\bar{L}_1$  is recursive and  $\bar{L}_2$  is recursively enumerable
  - $\bar{L}_1$  is recursive and  $\bar{L}_2$  is not recursively enumerable
  - $\bar{L}_1$  and  $\bar{L}_2$  are recursively enumerable
  - $\bar{L}_1$  is recursively enumerable and  $\bar{L}_2$  is recursive
107. Consider the languages

$$L_1 = \{WW^R \mid W \in \{0,1\}^*\}$$

$$L_2 = \{W\#W^R \mid W \in \{0,1\}^*\}, \text{ where } \# \text{ is a special symbol}$$

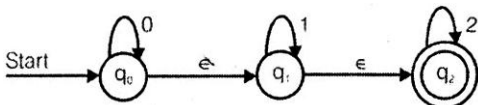
$$L_3 = \{WW \mid W \in \{0,1\}^*\}$$

Which one of the following is TRUE?

- $L_1$  is a deterministic CFL
- $L_2$  is a deterministic CFL
- $L_3$  is a CFL, but not a deterministic CFL
- $L_3$  is a deterministic CFL

**Directions for 108 and 109:**

Consider the following NFA with  $\epsilon$  moves.

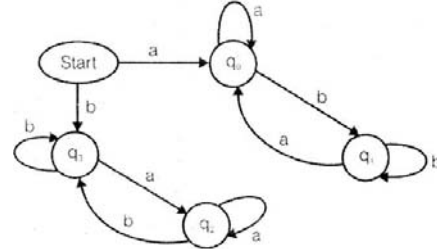


108. If the given NFA is converted to NFA without  $\epsilon$ -moves, which of the following denotes the set of final states?
- $\{q_2\}$
  - $\{q_1, q_2\}$
  - $\{q_0, q_1, q_2\}$
  - can't be determined
109. Which of the following strings will not be accepted by the given NFA?

- 001122
- 1122
- 21
- 22

**Directions for 110, 111 and 112:**

Consider the transition diagram of an DFA as given below:



110. Which should be the final state(s) of the DFA if it should accept strings starting with 'a' and ending with 'b'?
- $q_0$
  - $q_1$
  - $q_0, q_1$
  - $q_3$
111. Which of the following strings will be accepted if  $q_0$  and  $q_1$  are accepting states?
1. ababab
  2. babaaa
  3. aaaba
- 1, 2
  - 2, 3
  - 1, 3
  - None of these
112. Which of the following represents the set of accepting states if the language to be accepted contains strings having the same starting and ending symbols?
- $\{q_0\}$
  - $\{q_0, q_3\}$
  - $\{q_3\}$
  - $\{q_3, q_1\}$

**Directions for 113 and 114:**

Consider grammar  $G$

$$S \rightarrow AB, A \rightarrow aAA/\epsilon, B \rightarrow bBB/\epsilon.$$

113. Find the nullable symbols in the given grammar

- A
- B
- A, B and S
- A and B

114. If  $G_1$  is constructed from  $G$ , after eliminating  $\epsilon$ -productions, then  $G_1$  is given by

- $S \rightarrow AB$   
 $A \rightarrow aAA \mid \alpha A$   
 $B \rightarrow bBB \mid \beta A$
- $S \rightarrow AB \mid A \mid B$   
 $A \rightarrow aAA \mid \alpha A$   
 $B \rightarrow bBB \mid \beta B$
- $S \rightarrow AB$   
 $A \rightarrow aAA \mid \alpha A \mid \alpha$   
 $B \rightarrow bBB \mid \beta B \mid \beta$
- $S \rightarrow AB \mid A \mid B$   
 $A \rightarrow aAA \mid \alpha A \alpha$   
 $B \rightarrow bBB \mid \beta B \mid \beta$

**Directions for 115, 116 and 117**

The 'value' of a R.E.  $r$  over  $\Sigma$ , denoted by  $\text{val}(r)$ , is defined as follows:

- $\text{val}(\phi) = 0$
- $\text{val}(\epsilon) = 0$
- $\text{val}(a) = 0, \forall a \in \Sigma$
- $\text{val}((rs)) = \text{val}((r+s)) = \max(\text{val}(r), \text{val}(s))$
- $\text{val}((r^*)) = \text{val}(r) + 1$

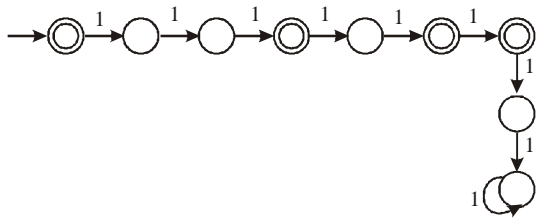
Find the value of the following R.E.

115.  $(a(a + a^*aa))$
- 0
  - 1
  - 2
  - None of these
116.  $(a(a + a^*aa) + aaa)^*$
- 1
  - 2
  - 3
  - 4
117.  $(a^*b^*b^*b^*)$  where  $\Sigma = \{a, b\}$
- 2
  - 3
  - 4
  - none of these



# HINTS & SOLUTIONS

## PAST GATE QUESTIONS EXERCISE

1. (a)
2. (b) In all the options there is linear relationship among strings so all *CFL*'s, but  $L_1$  &  $L_3$  can be accepted by *PDA*,  $L_2$  can be accepted by deterministic *CFL* due to presence of special symbol  $\$$  which tells the middle of the string, so deterministic.
3. (b) The rules here used will be.  
All those languages which are recursive their complements are also recursive.  
So option (A) & (B) can be correct.  
Now languages which are recursively enumerable but not recursive, their complements can't be recursively enumerable.  
So only option (B) is correct  
Hence (B) is correct option
4. (a)  $L_1$  and  $L_2$  are context-free languages and therefore  $L_1 \cap L_2$  may or may not be context-free, since *CFL*s are not closed under intersection. Now, let us look at  $L_1 \cap L_2$ .  
 $L_1 \cap L_2 = \{a^n b^n c^n \mid n > 0\}$   
which is clearly not context-free but is context sensitive.
5. (d) We know that, the languages accepted by non-deterministic finite automata are also accepted by deterministic finite automata. This may not be in the case of context-free languages.  
Therefore,  $D_f = N_f$  and  $D_p \subset N_p$
6. (b)  $a$  is followed by two or more than  $2b$ 's so the language recognized by  $M$  is  $\{W \in \{a, b\}^* \mid \dots\}$ .
7. (c) Option (c)  
According to the option,  $P_2$  is reducible to  $P_3$ .  
Also it gives that  $P_2$  is undecidable.  
From (a) and (b);  
 $P_3$  is undecidable.
8. (d) Given language  $L = (1111 + 11111)^*$ . Here, we can see that more than eight 1's can be generated by the combination of 111 and 11111. Therefore, number of states =  $8 + 1 = 9$ .  
These can be represented as  

9. (b)  $L_1$  is regular language  
 $L_2$  is *CFL*.  
 $L_3$  is recursively enumerable but not *REC*.

- (A)  $L_1 \cap L_2$  is *CFL* is true  
(B)  $L_3 \cap L_1$  is recursive, not necessary so false.  
(C) & (D) are also true.
10. (b) Due to  $S \rightarrow S$  this Grammar is ambiguous right hand side has two Non terminals. Also the strings like  $aaabbb$  have equal no. of  $a$ 's &  $b$ 's but can't be produced by this grammar. So 2 is false.  
Statement 3 is true since it is a *CFG* so accepted by *PDA*.
11. (a) There is a difference between  $SHAM_3$  and  $DHAM_3$  that  $SHAM_3$  is used for finding a Hamiltonian cycle in a graph but  $DHAM_3$  is the problem of determining if a graph exists in a Hamiltonian cycle. Also it is given that  $|V|$  is divisible by 3, hence the problem can be reduced from 3 SAT which further determines that  $SHAM_3$  and  $DHAM_3$  are NP hard.
12. (d) Given that,  $d(s) \bmod 5 = 2$  and  $d(s) \bmod 7 \neq 4$   
Both the languages are regular languages, since there exists deterministic finite automata for both of them. We also have an algorithm to check the regular nature of the language therefore  $L$  is regular.
13. (c) Option (A), (B) & (D) can be accepted by *DFA*, & there is no linear relationship between the no. of 0's & 1's in the string but in (C)  $n_0(S) - n_1(S) \leq 4$  can't be accepted by *DFA*, we require a *PDA*.  
So not regular.
14. (d) The language is context free or not, this can be proved by finding out the grammar for all the languages.

$$L_1 = \{0^{n+m} 1^n 0^m \mid n, m \geq 0\}$$

The production for  $L_1$  as follows:

$$S \rightarrow 0S_0 \mid 0A_1 \mid \epsilon$$

$$A \rightarrow 0A_1 \mid \epsilon$$

Now, we need to apply these values of the production individually to generate

$$0^{n+m} \mid 1^n 0^m$$

that is,

$$0^m 0^n 1^n 0^m \rightarrow \text{To prove}$$

Applying  $0S_0$  ( $m$  times)

$$S \rightarrow 0S_0, S \rightarrow 0^m S_0^m$$

Applying,  $S \rightarrow 0A_1$

$$\rightarrow S \rightarrow 0^m 0A_1 0^m$$

Here applying,  $A \rightarrow 0A_1$ ,  $n - 1$  times

$$S \rightarrow 0^m 0 0^{n-1} A_1^{n-1} 10^m$$

$$\rightarrow S \rightarrow 0^m 0^n 1^n 0^m$$

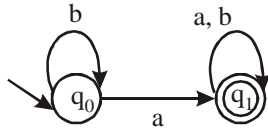
Hence, proved, so  $L_1$  is context-free.

Going through the same procedure, we can see that two comparisons are made in the  $L_2$  language. So it is not context-free.

$L_3 \rightarrow$  Going through the same procedure again, we can see that two comparisons are made in the  $L_3$  language, so it is not context free.

15. (a)  $S$  is  $NP$  complete and a  $NP$  complete problem is reducible to some unknown problem then that problem is also  $NP$  complete. So  $S \rightarrow \Delta, R$  the  $R$  is  $NP$  complete.
16. (b) The FSA as obtained in the previous question is  $b^* a (a + b)^*$

The minimum number of states are thus given by



$q_0$  and  $q_1$  are the state: that are required at most and hence the minimum number of states is 2 ( $q_0$  and  $q_1$ ).

17. (c) The behaviour as per the given PDA is as seen below  
 $q_0$  to  $q_1 \rightarrow b^*a$   
 $q_1$  to  $q_2 \rightarrow (a + b)^*$   
 Regular expression =  $b^* a (a + b)^*$

18. (c) Lets study the regular language.  
 Conventions on regular expressions
1. Bold face is not used for regular expressions when the expression is not confusing. So, for example,  $(r + s)$  is used instead  $(r + s)$ .
  2. The operation  $*$  has precedence over concatenation, which further has precedence over union  $(+)$ . Thus, the regular expression  $(a + b(c^*))$  is written as  $a + bc^*$ .
  3. The concatenation of  $k$   $r$ 's, where  $r$  is regular expression, is written as  $r^k$ . Thus, for example  $rr = r^2$ . The language corresponding to  $r^k$  is  $L_r^k$ , where  $L_r$  is language corresponding to the regular expression  $r$ . For a recursive definition of  $L_r^k$ .
  4. The  $(r^+)$  is used as a regular expression to represent  $L_r^+$ .

Since, language  $L$  can be expressed as

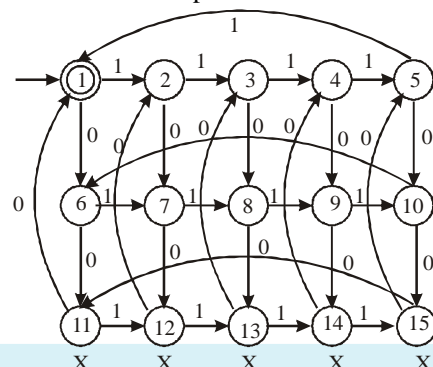
$$r = [0(0+1)^*0] + [1(0+1)^*1]$$

and follows the above convention, therefore is regular

19. (b)  $L = \{0^i 2^j / i > 0\}$ , this language can't be accepted by  $DFA$  to regular, but it is recursive & can be accepted by  $PDA$  to  $CFL$ .
20. (a) It is given that the 0's and 1's are divisible by 3 and 5 and we know that 3 and 5 do not have any factor other than themselves or 1 i.e., these cannot be further breakdown.

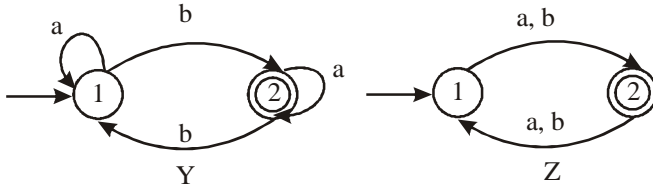
Therefore, number of states =  $3 \times 5 = 15$

The schematic representation is as follows:



1. single-terminal
  2. non-terminal followed by a terminal
  3.  $\lambda$
- Statement 4 False: The derivation trees in CNF are binary trees.

26. (a) **Given:** Transition table for Y and Z  
The number of states of Z = 2  
The number of states of Y = 2  
 $Z \times Y$



No. of states of the product of Z and Y =  $2 \times 2 = 4$   
Now, the states as per the given options are P, Q, R and S. The finite state automata is

	a	b
$\rightarrow P$	S	R
Q	R	S
R(F)	Q	P
S	Q	P

Table for  $Z \times Y$  is

	a	b
$\rightarrow (1, 2)$	(2, 1)	(2, 2)
(1, 2)	(2, 2)	(2, 1)
(2, 1)	(1, 1)	(1, 2)
F(2, 2)	(1, 2)	(1, 1)

27. (d) Option (a) True: Non-deterministic finite automata can be converted into the deterministic finite automata.  
Option (b) True: With reference to option (a), same is with the non-deterministic turing machine.  
Option (c) True: Regular language is always context-free but the reverse is not true.  
Option (d) False: We know that a set is a subset of itself and hence, every subset of recursively enumerable set is not recursive.
28. (b) As per the theorem, a language is recursive if that language and its complement are recursively enumerable. Therefore, L is recursive.
29. (b) **Statement 1 Decidable:** The algorithm can be used to check the finiteness/infiniteness on the DFA and also the two given DFAs, a product DFA can be constructed.

**Statement 2 Undecidable:** It is not decidable since the language is context-free.

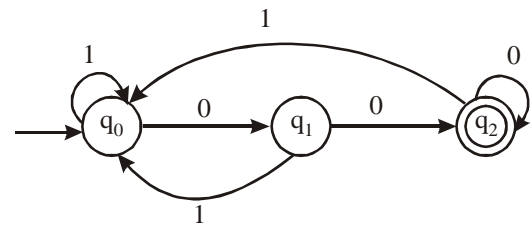
**Statement 3 Undecidable:** It is also undecidable that whether two push-down automata accept the same language.

**Statement 4 Decidable:** If the LHS of each production has one and only one variable then it is a context-free grammar.

30. (d) (d)  $\{a^p \mid p \text{ is a prime no.}\}$

This prime no. is extra constraint so this language is neither LFG nor RG but it can be accepted by turing machine.

31. (c)



State	Input	Output
$q_0$ to $q_0$	1, 0	00
$q_1$ to $q_0$	1, 0	00
$q_2$ to $q_0$	1, 0	00

Therefore, the above DFA ends with 00.

32. (c) It is given that  $L = L_1 \cap L_2$

Let's first analyze the given language and check whether it is context-free or not. The context-free languages are defined as below—

A context-free language is  $L = \{a^n b^n : n \geq 1\}$  that is

the language of all non-empty even-length strings. It consists of

1. the entire first halves of which are a's.
2. the entire second halves of which are b's.

L is generated by the grammar  $S \rightarrow aSb \mid ab$ , and is accepted by the push-down automaton  $M = (\{q_0, q_1, q_f\}, \{a, b\}, \{a, z\}, \delta, q_0, \{q_f\})$  where  $\delta$  is defined as

$\delta(q_0, a, z) = (q_0, a)$

$\delta(q_0, a, a) = (q_0, a)$

$\delta(q_0, b, a) = (q_1, x)$

$\delta(q_1, b, a) = (q_1, x)$

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

$\delta(\text{state}_1, \text{read pop}) = (\text{state}_2, \text{push})$

where z is initial stack symbol and x means pop action.

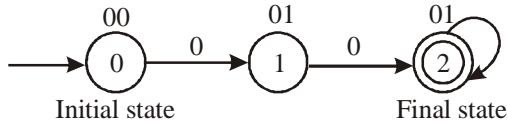
Here, in we are given, where a and b are of equal lengths

and followed by c which is of different length. This definitely shows that the languages are context-free but not regular.

33. (a) The initial state = 00

Final state required = 01

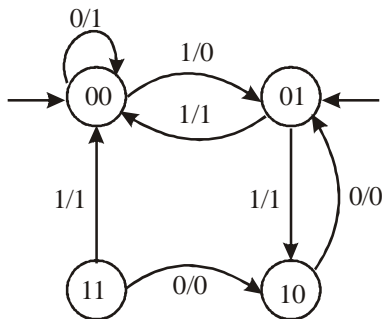
Let us construct the transition diagram for the given.



We get the total number of states to be 3 when getting the desired output. The transition diagram can further be elaborated as below.

There can be 4 states 00, 01, 10, 11.

With this, the FSM can be designed as



The desired output is obtained with the input string 101, however, the concern is number of states which we found to be 3.

34. (b) Regular expressions are meant for lexical analysis to define tokens. Pushdown Automata is used to accept context free language which are used for syntax analysis.

Data flow analysis is a technique for code optimization.

Register allocation is used for code generation.

So  $P-3, Q-1, R-4, S-2$ .

35. (d) (a) true, since minimal *DFA* for every regular language is possible.

(b) true, *NFA* can be converted into an equivalent *PDA*.

(c) *CG'S* are not recursive but their complements are.

(d) false, since non deterministic *PDA* represents, non deterministic

*CFG*, since *NDCFG* and *CFG* are proper subsets so conversion required.

36. (c) We are given with the relation

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

Here, the accepting languages are

$$L = \{00, 000, 100, 001, 010, 0000, 0001, 1000, 1001, 0100, 1100, 0010, 0011, 0110, 0101, 1010, \dots\}$$

The common feature in the accepting languages can be seen that they consist of atleast two 0's.

37. (b) Given grammar  $S \rightarrow aSabSbab$ .

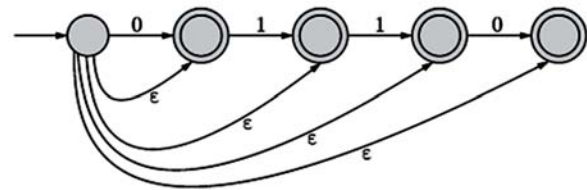
The strings generated through this grammar is definitely palindromes, but not all it can only generate palindromes of odd length only so (A) & (D) are false, (B) is correct.

Also it can generate palindromes which start and end with same symbol, but not all strings eg. *aabababba*.

38. (c)  $L$  is the set of all substrings of  $w$  where  $w \in \{0,1\}^*$

Any string in  $L$  would have length 0 to  $n$ , with any no. of 1's and 0's

The *NFA*



Here  $n = 4$

So to accept all the substrings the no. of states required are  $n + 1 = 4 + 1 = 5$

Hence (c) is correct option.

39. (d) These sort of languages are accepted by *PDA*, so all should be context free languages.  $L_2$  &  $L_3$  are definitely *CFL* since accepted by stack of *PDA*.

And also  $L_1$  &  $L_4$  are linear comparisons of  $i$  &  $j$  so can also be represented using *PDA*. So all are context free languages.

40. (b) It is given that  $L$  is the set of all bit strings with even number of 1's so the regular expression should exhibit the same.

Now, the min string should be  $\epsilon$  and the string should be  $\epsilon, 0, 11, 101, \dots$

The string obtained from such expression is  $0^* (10^* 10^*)^*$

41. (b)  $L_1 \rightarrow$  recursive

$L_2, L_3 \rightarrow$  recursively enumerable but not recursive.

So  $L_1$  can be recursive enumerable.

$RE - RE = RE$

So  $L_1 - L_3$  is recursively enumerable.

42. (c)  $L_1 = \{0^p 1^q \mid p, q \in \mathbb{N}\}$  is regular language

$L_2 = \{0^p 1^q \mid p, q \in \mathbb{N} \text{ and } p = q\}$  is context-free

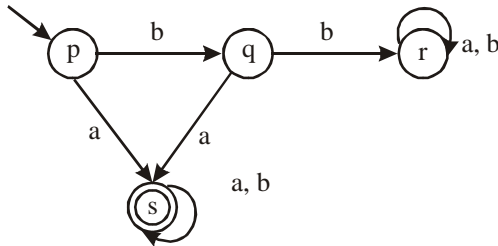
language

$L_3 = \{0^p 1^q 0^r \mid p, q, r \in \mathbb{N} \text{ and } p = q = r\}$  is not

context-free.



43. (a) As state (s) and (t) both are final states and accepting  $a^* + b^*$ , we can combine both states and we will get



44. (b) As  $n$  is constant atleast  $n + 1$  states will required to design  $a^{nk}$ .
45. (b) DPDA and NPDA because an NPDA cannot be converted into DPDA.
46. (c)  $\Sigma^* - P$  and  $\Sigma^* - P$  is the complement of  $P$  and complement of regular language is also regular.

47. (b)

ab	00	01	11	10
cd				
00	1	X	X	1
01	X			1
11				
10	1			X

$$b'c' + b'd' \text{ or } \overline{bc} + \overline{bd}$$

48. (b) Given, there are  $n$  strings and length of each string is  $n$ . Normal recurrence relation for merge sort is

$$T(n) = 2T(n/2) + Cn$$

Here, we need  $n$  comparisons to compare two strings i.e., 2 elements of set.

So, recurrence relation will be

$$T(n) = 2T(n/2) + Cn^2$$

By solving this using master method, we will get

$$T(n) = O(n^2)$$

49. (c) Time complexity to search an element in a balanced binary search tree is

$$= \log [\text{number of elements in binary search tree}]$$

$$= \log [n \times 2^n]$$

$$= \log n + \log 2^n$$

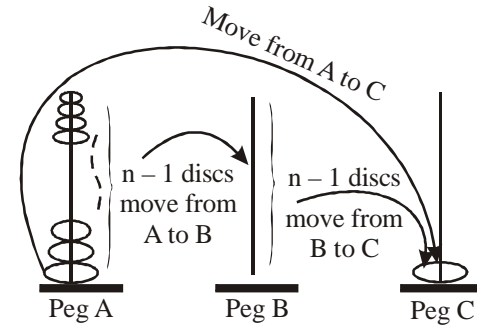
$$= \log n + n \log 2$$

$$= \log n + n \quad (\because \text{while } \log_2 2 = 1)$$

$$= \theta(n)$$

50. (d) Tower of Hanoi problem with  $n$  discs are follows:  
We have given  $n$  discs in peg A and we have to move all  $n$  discs from A to C peg. We will take help of extra peg B and we arrange all the  $n$  discs in peg C as follows

that the smallest disc is at the top position and the largest disc is at the lowest position.



For this, we can recursion algorithm. Firstly, we have to move all upper  $n - 1$  discs from peg A to B, then last largest disc moved from peg A to C, then again  $(n - 1)$  discs moved from peg B to C.

#### Algorithm

function  $(n, A, B, C)$

```
{
function  $(n - 1, A, C, B);$ 
 $A \rightarrow C$ 
function  $(n - 1, B, A, C);$ 
}
```

Then, the recurrence relation for this problem is

$$T(n) = 2T(n - 1) + 1$$

51. (c) As we have given,

$A(n) \rightarrow$  Average case complexity

$W(n) \rightarrow$  Worst case complexity

As we know that average case will always be less than or equal to worst case complexity.

$$A(n) \leq W(n)$$

Consider option (a):

$$A(n) = \Omega(W(n))$$

$\therefore$  It says  $W(n)$ , worst case complexity is less than the average case complexity which is wrong because  $\Omega$  asymptotic notation shows  $(\geq)$  sign. Hence, this option is false.

Consider option (b)

$$A(n) = \theta(W(n))$$

$\therefore$  It says  $A(n)$ , worst case complexity is same as worst case complexity because  $\theta$  notation shows equality sign. Hence, this is false.

Consider option (c)

$$A(n) = O(W(n))$$

$\therefore$  It says that average case complexity  $A(n)$  is less than or equal to  $W(n)$  worst case complexity. Hence, this option is correct.

Consider option (d)

$$A(n) = O(W(n))$$

$\therefore$  It says that  $A(n)$  is strictly less than  $W(n)$ , so this option is false.

280

52. (a) Considering the languages  $L_1 = \phi$  and  $L_2 = \{a\}$   
 For all languages  $L$  it is known that  $\phi \cdot L = \phi$

Suppose,  $\exists a$  string  $s \in \phi \cdot L$

$\exists s'$  such that  $s = s' \cdot s''$

and  $s' \in \phi \cdot s'' \in L$

But  $s' \in \phi \quad \therefore \phi$  is an empty language.

$\therefore \phi \cdot L = \phi$

$\{\epsilon\} \subseteq L^*$  for all languages  $L$ .

As it means taking letters from the language and concatenating them '0' times creating a '0' length string which is possible for all languages.

$\therefore \epsilon$  is in  $\phi^*$

$L_1 \cdot L_2 = \phi$

and  $L_1 = \{\epsilon\}$

$L_1 \cdot L_2^* \cup L_1^* = (L_1 \cdot L_2) \cup L_1^*$

$\therefore \phi \cup \{\epsilon\} = \{\epsilon\}$

53. (c) 1. Non-deterministic Turing Machine can be simulated by a deterministic Turing Machine with exponential time true.

2. Turing recognizable language are "not" closed under complementation. For any Turing recognizable language the Turing Machine 'T' recognizing 'L' may not terminate on inputs

$x \notin L$  - False

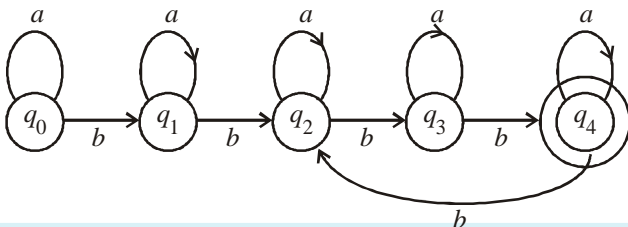
3. Turing decidable languages are CLOSED under union and complementation. It is easy to determine if Turing machine is decidable-True

So, answer is option (c) only 2.

54. (d) 1. G is a CFG. Is  $L(G) = \phi$ ?  
 Decidable  
 2. G is a CFG. Is  $L(G) = \Sigma^*$ ?  
 Undecidable  
 3. M is a Turing Machine. Is  $L(M)$  regular?  
 Undecidable.  
 4. A is a DFA and N is an NFA. Is  $L(A) = L(N)$ ?  
 Decidable.

Hence, the correct answer is (d) 2 and 3 only.

55. (c) We know that a language L is regular if an equivalent finite Automaton can be constructed for it.  
 DFA can be constructed for L as follows:-  
 Note that L contains the strings that has  $3k + 1$  number of 'b's (that is 4, 7, 10, .....n ..... no. of 'b's and any no. of 'a's)



$q_0$  is the initial state.  $q_4$  is the final state.

FA state remains same when input symbol is "a" at any point of time.

When 1st 'b' is read, state is changed from  $q_0$  to  $q_1$ .

When 2nd 'b' is read, state is changed from  $q_1$  to  $q_2$ .

When 3rd 'b' is read, state is changed from  $q_2$  to  $q_3$ .

& when 4th 'b' is read, state is changed from  $q_3$  to  $q_4$ .

If no more 'b' is encountered, the string is accepted when last input symbol is read.

However if 5<sup>th</sup> "b" is there is string, state is changed from  $q_1$  to  $q_2$  so that to accept the string 2 more bs must be there in the string at least.

Continuing in this way only those strings are accepted that has  $3k + 1$  ( $k \in N$ ) number of 'b's. (i.e. 4, 7, 10 ... number of 'b's).

56. (a) Following paths can be taken by the finite Automaton for the input string "0011":—

$q_0 \xrightarrow{0} q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_0 \xrightarrow{1} q_0$

Reachable states

$\{q_0\}$

$q_0 \xrightarrow{0} q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_0 \xrightarrow{1} q_1$

$\{q_0, q_1\}$

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

$\{q_0, q_1, q_2\}$

We note that no other path is possible for the input string "0011".

So, finally union of all three cases gives us the set of Reachable states which is  $\{q_0, q_1, q_2\}$

57. (c) Both L and  $\bar{L}$  are recursively enumerable but not recursive.

Set of recursive languages is subset of the set of recursively enumerable languages.

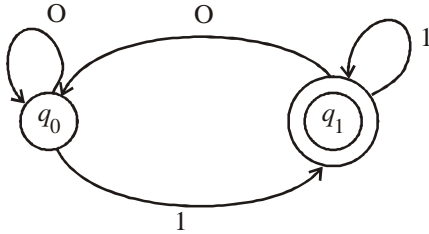
So, if a language is recursive, It must be R.E. also.

- (A) May be true as a language L and its complement  $\bar{L}$  need not be recursively enumerable.  
 (B) May be true if L is r.e. but not recursive and  $\bar{L}$  is not recursively enumerable  
 (D) May be true as L and  $\bar{L}$  both can be recursive.

However (C) is not possible because if Both L and  $\bar{L}$  are recursively enumerable then by a well known theorem of complexity theory either L or  $\bar{L}$  has to be recursive.

58. (d) I, II and III.

Given DFA:



(I)  $0^*1(1+00^*1)^*$  it represents the given DFA using following transition:

$0^* :- 1 \ q_0 \rightarrow q_0$

$1 :- q_0 \rightarrow q_1$

$(1+00^*1)^* :- q_1 \rightarrow q_0 \rightarrow q_0 \rightarrow q_1$

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

So, the string is accepted.

(II)  $0^*1^*1+11^*0^*1 :-$

Following transition occurs :-

$0^* :- q_0 \rightarrow q_0$

$1^* :- q_0 \rightarrow q_0$  (if 1 is absent) or  $1^* :- q_0 \rightarrow q_1$  (if 1 is present)

$(1+11^*) : q_0 \rightarrow q_1 \quad q_1 \rightarrow q_1$

$0^* :- q_1 \rightarrow q_1$  (if 0 is absent)  $q_1 \rightarrow q_0$  (if 0 is present)

$1 :- q_1 \rightarrow q_1 \quad q_0 \rightarrow q_1$

final state is  $q_1$ , this string is also accepted.

(III)  $(0+1)^*1 :-$  Following transition occurs :-

$(0+1)^* :- q_0 \rightarrow q_0 \rightarrow q_1$

$(0+) \ (1)$

$1 :- q_1 \rightarrow q_1$

Ans final state is  $q_1$ , so this string is also accepted. Hence

(d) is true.

59. (a)

60. (d) If B is not recursively enumerable then A need not be recursively enumerable.

61. (b) The language is recursive enumerable and it is undecidable

62. (a)  $L_1$  is regular but  $L_2$  is not

63. 3

$a^*b^*(ba)^*a^*$

Length 0 is present (as it accepts  $\epsilon$ )

Length 1 is present (a, b)

Length 2 is present (aa, ab, ba, bb)

Length 3 is not present (bab not present)

$\therefore$  it is 3

64. (c)  $\Sigma^*$  is countably finite

$2\Sigma^*$  is power set of  $\Sigma^*$

The powerset of countably infinite set is uncountable

$\therefore 2\Sigma^*$  is uncountable and  $\Sigma^*$  is countable.

## PRACTICE EXERCISE

1. (c) Regular and context free languages are both closed under union and concatenation while only regular is closed under intersection.

$\therefore$  Ans is (c)

2. (d)  $ac \in L(r_1)$ , since we can take  $b^*$  as  $\epsilon$  and  $c^*$  as  $c$

$ac \in L(r_3)$ , since  $(a \cup b \cup c)^*$  includes all combinations of a, b, c.

$ac \notin L(r_2)$  since when ever  $(a^*b+c)^*$  is taken to include a, "a is always followed by b".

$a^*b = b, ab, aab, \dots$  & so on.

$\therefore$  Ans is (d)

3. (d)  $L(r_1)$  is the Language starting with 'a'.

$L(r_2)$  is the Language starting with 'b'.

Since any word belonging to  $\Sigma^*$  either starts with "a" or starts with "b" or is " $\lambda$ ", therefore

$$L(r_1) \cap L(r_2) \cap \{\lambda\} = \Sigma^*$$

So, Ans is (d)

4. (d) Note that

$$(b^*a^*) = (a^*b^*)^* = (a^* + b^*)^* = (a + b)^*$$

so for (a)  $L((b^*a^*)^*(d^*c^*)^*)$  gives all the combinations of a and b followed by all the combinations of c & d.

$$\text{for (b) } L((d^*c^*b^*a^*)^*) = (a + b + c + d)^*$$

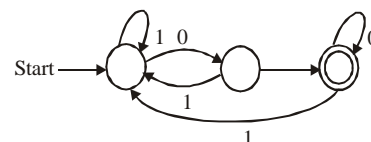
$$\text{for (c) } L((a^*b^*a^*c^*d^*)^*) = (a + b + c + d + a^*)^* = (a + b + c + d)^*$$

5. (b) Choice 'a' is incorrect since it does not include the string "a", "b" and " $\lambda$ " (all of which do not end with ab).

None of choices 'c' or 'd' accept the string 'a', So they can't be represent specified language.

6. (a) For DFA accepting all the strings with number of 'a' divisible by 4, four states are required similarly for DFA accepting all the strings with number of b's divisible by 5, five states are required and for their combination, states will be multiplied So  $5 \times 4 = 20$  states will be required.

7. (b) The DFA will be



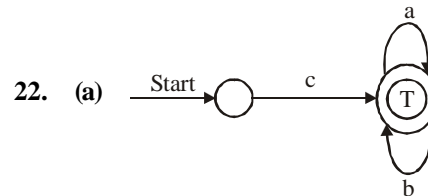
For these kind of problems the required number of states are always equal to "the length of the string that it is ending with" + 1

282

8. (c) Since  $L = X^* = \{0, 1\}^* = (0 + 1)^*$  &  $R = \{0^n 1^n \mid n > 0\}$ , LUR produces  $(0 + 1)^*$  which is regular language and R is not regular as there is no regular expression for that.
9. (b) Since  $L_1$  and  $L_2$  both are accepted by deterministic PDA (in both cases, no guessing is required for the push and pop operations while operating the stack) since these languages are accepted by DPDA, it will also be accepted by NPDA.
10. (b) Regular languages are closed under union, intersection, complementation, Kleene closure, concatenation. According to closure properties so all of a, b, c, & d are regular. Notice that,  $S^* - L_1 = L_1^c$  so, set  $L_1 \cap L_2$  is also regular.
11. (d) All of the above functions are total recursive functions. The total recursive functions are like recursive language. Turing machine halts on each and every input of recursive language. All of the above functions are Turing computable.
12. (c)  $L_1 = \{a^n b^n c^n, n \geq 0\} = \{\lambda, abc, a^2 b^2 c^2, \dots\}$   
 $L_2 = \{a^{2n} b^{2n} c^{2n}, n \geq 0\} = \{\lambda, a^2 b^2 c^2, a^4 b^4 c^4, \dots\}$   
 $L_3 = \{a^{2n} b^{2n} c^n, n \geq 0\} = \{\lambda, a^2 b^2 c^2, a^4 b^4 c^2, \dots\}$   
as we can easily see that  
(i)  $L_1$  contains all the words generated by  $L_2$  and also it contains some extra strings also.  
 $\therefore L_1 \supseteq L_2$  (or  $L_2 \subseteq L_1$ )  
(ii) Since only  $\lambda$  is common in  $L_2$  and  $L_3$   
Hence  $L_2 \not\subseteq L_3$
13. (a) 1.  $0^* 1^*$  does not ensure at least one 0 at the beginning and one 1 at the end.  
2.  $00^*(0 + 1)^* 1$  ensures the specified condition.  
3.  $0(0 + 1)^* 1$  ensures the specified condition.
14. (b) The given grammar is context-free. It is not proper since it includes an  $\epsilon$ -production. A typical derivation in this grammar is  
 $S \rightarrow aSa \rightarrow aaSaa \rightarrow aabSbaa \rightarrow aabbbaa$ .  
This makes it clear that  $L(G) = \{ww^R : w \in \{a, b\}^*\}$ .  
The language is context-free.
15. (d) According to the definition of Moore machine (d) is the only correct option.
16. (b) 1.  $r_1 = \epsilon$  is a regular expression representing a set  $\{\epsilon\}$ . It is a regular expression.  
2.  $r_2 = 0^* 1^*$  is a regular expression.
17. (b) Since  $L_1$  and  $L_2$  are regular and context free languages respectively CFL are closed under regular  $\cup, \cap$ , therefore

- (a)  $L_1 \cup L_2$  is CFL  
(b)  $L_1 \cap L_2$  is not regular  
(c)  $L_1^*$  is regular  
Hence the correct Ans. is (b)

18. (b) Since  $\Sigma^*$  is all the combinations of 0 and 1 and yet it is enumerable and belongs to the category of countably infinite. But for  $2^{\Sigma^*}$  which is the power set of  $\Sigma^*$  which is infinite. By diagonalisation argument, power set of any countably infinite set is always uncountable, hence  $2^{\Sigma^*}$  belongs to uncountably infinite set.
19. (a) Here for the languages (i) and (ii) we can design a PDA. Hence they are CFL.  
In (iii) and (iv) we cannot conclude that  $no(a) = no(c)$  and  $no(b) = no(d)$  using only one stack memory so they can't CFL.
20. (c) Only option (c) satisfies the given machine, hence  $L(M) = \{\text{words containing aa or bb as a subword}\}$
21. (b) Binary no. that are multiples of two is  
 $L = \{0, 10, 100, 1000, 110, 100, \dots\}$   
i.e. strings ending with 0  
 $\therefore$  (b)  $(0/1)^* 0$  is the correct answer.



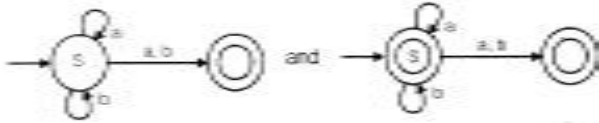
The language generated by the above DFA is starting with c followed by any no. of a's & b's.

$$\text{Hence } L = c(a \cup b)^*$$

23. (b) Since  $L_1$  cannot be recognized by PDA but  $L_2$  can be recognized by a PDA.  $L_1$  is not context free but  $L_2$  is context free.  
For  $L_2 = \{a^n b^m c^m d^n \mid n \geq 1, m \geq 1\}$   
a's are pushed into stack, then b's are pushed then b's are popped from the stack checking with each arrival of corresponding c. After all b's are popped and if no. of b's popped = no. of c's arrived then, as a will be at top of stack, when the first d arrives.  
So, no of b's = no. of c's  $[m = m]$ . Similarly when d's arrive, we can check that  
No. of a's = No. of d's  $[n = n]$ , by popping an a for every d and checking that at end of input stack is empty. This can't be done in the case of  $L_1$ , since after pushing a's & then b's, when c's appear in input, they have to be compared with a's which are unfortunately at the bottom of the stack and cannot be popped to do a comparison.

24. (b) The DFA's construction for the grammars

$S \longrightarrow aS \mid bS \mid a \mid b$  and  $S \longrightarrow aS \mid bS \mid a \mid b \mid \lambda$  are respectively.

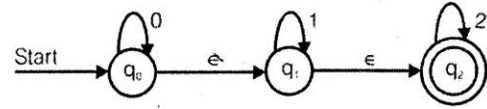


The required languages are

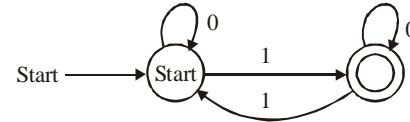
$$L_1 = (a + b)(a + b)^*$$

$$L_2 = (a + b)^*$$

25. (a) ATM cannot solve the halting problem because it cannot find in finite time that for the given word the machine will halt or not.
26. (d) Since every language can be represented through regular expression and its equivalent machine. Also for every deterministic machine there exist its equivalent non-deterministic machine.
27. (c) The FSM shown accepts only  $\epsilon$ . Also it is the starting as well as final state and no other input is there.
28. (b) The FSM shown has no final states to accept strings. Hence, it accepts no strings. Hence, option (b) is correct.
29. (d) It follows from the closure properties of CFL's. CFL's are closed under  $\cup$ , concatenation &  $*$ , but not under  $\cap$  and  $L^C$ .
30. (b) For set  $A = \{a^n b^n a^n \mid n = 1, 2, 3, \dots\}$  the no. of comparisons are 2. first one for checking No. of  $a$  = No. of  $b$  and, second one for again checking No. of  $b$  = No. of  $a$ 's following the  $b$ 's. As we know, a pda can do only one of these comparisons. Hence the given language is not context free.
31. (d) Here  $L_1 = (a + b)^* a$  is nothing but set of all strings of  $a$  and  $b$  ending with  $a$ .  
 $L_2 = b(a + b)^*$  is set of all strings of  $a$  &  $b$  starting with  $b$ .  
 The intersection of  $L_1$  and  $L_2$  is set of all strings starting with  $b$  and ending with  $a$ .  
 $\therefore L_1 \cap L_2 = b(a + b)^* a$
32. (d) Here
- (a) is not correct because we can't find the middle of palindrome, without guessing.
- (b) is also not correct because of same reason.
33. (b) This is a standard DFA for all strings ending with "a". Hence the language is  $(a + b)^* a$ .
34. (b) Regarding the power of recognizing of languages
- (a) NFA has same power as DFA
- (b) NDTMs are equivalent to DTMs
- (c) Multiple tape TMs are equivalent to single tape TMs
- Only DPDA  $\subseteq$  NPDA in the power.
35. (a) Since  $\forall w \in \Sigma^*$  whether or not  $w \in L$ , so  $L$  is decidable. can be determined in finite time.

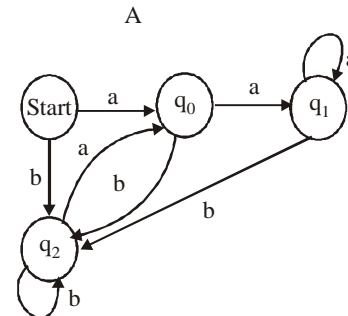


36. (d) The given FA accepts all combinations of even number of 0's and 1's.
37. (a) The automation given below, accepts the words with odd of 1's.

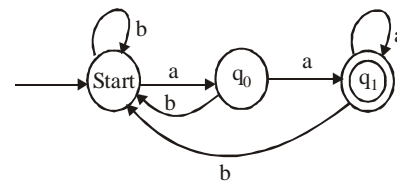


clearly, the automation requires only 2 states. Hence the correct option is (a).

38. (d) The unary number has a single symbol and the length of the string is its value. So we have to accept strings where, no of 1's divisible by 3. On first symbols of number (i.e.), we will move to remainder 1 state, ( $q_1$ ) when next input is received we move to remainder 2 state ( $q_2$ ) and on next we move to 0 remainder state (i.e.  $q_3$ ) as shown in the choice (1).
39. (b) Drawing the FSM for the given transition diagram we can clearly see that if ' $q_1$ ' is the final state then only the given FSM can generate words which are all ending with "aa".

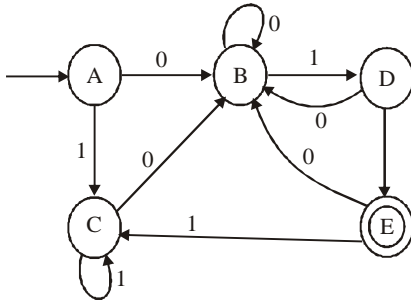


40. (c) The minimum state DFA accepting the above language is



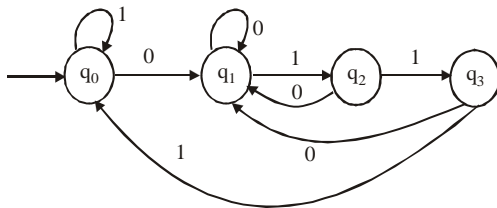
41. (d)  $(a + b)^*$  means all combination of 'a' and 'b'. Hence 'a' in  $(a + b)^* a(a + b)^*$  signifies that at least one 'a' must be there.
42. (d)
43. (c) Clearly 0's followed by 1's followed by 2's being accepted so choice (c)  $0^* 1^* 2^*$  is correct.
44. (b)  $(a + b)(a + b)$  generates all the 2 length strings  $aa$ ,  $ba$ ,  $ab$  &  $bb$ . So  $[(a + b)(a + b)]^*$  will generate all the even length strings, including  $\lambda$ .

45. (c) Drawing the dfa corresponding to the given transition table.



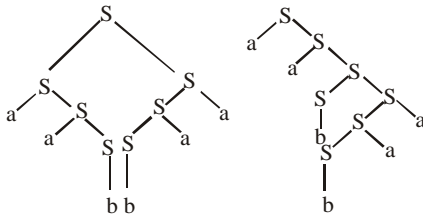
Use dfa minimisation algorithm

The corresponding minimum state dfa for this language can be constructed as



Only 4 states are needed.

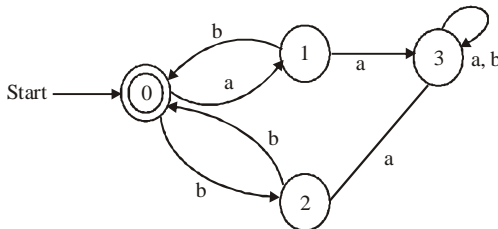
46. (a) Two derivation trees are possible for aabba as given below



Therefore (i) and (ii) are both true.

Choice (iii): 'aba' is also accepted by the given grammar. Therefore (iii) is false.

47. (b)



We can clearly see that any no. of loops of "ab" and "bb" can be accepted and '0' is the accepting state.

48. (c) The given regular expression  $00(1 + 10)^*$  is not complete to denotes all strings of 0's and 1's with atleast two consecutive 0's.

The correct regular expression must be  $(0 + 1)^*00(0 + 1)^*$

49. (a) Deterministic PDA can accept  
(b) The set of all strings of balanced parenthesis.

- (c)  $L = \{WcW^R/W \text{ in } (0 + 1)^*\}$

- (d)  $L = \{0^n 1^n | n \geq 0\}$

in each of the above the PDA can deterministically separate the push and pop operations.

But it cannot do the same in case (a), since it has to guess middle of palindrome, In DPDA we can't do guessing. NDPA is required for this.

50. (b) The transition function

$$\delta(q_0, 0, z_0) = (q_0, xz_0)$$

represents that when input is 0, stack top is  $z_0$  and current state is  $q_0$  than the stack top becomes  $xz_0$  and state remains at  $q_0$ .

The given set of transition functions checks for number of 0's followed by No. of 1's. Whenever a "0" comes, stack top become x and when the "1" come it is removed from stack Top. If at the end No input is left and stack is empty than the word is accepted.

Hence, (b)  $L = \{0^n 1^n | n \geq 1\}$  is correct

51. (a) It is possible to have a subset of regular language which is not regular and a subset of CFL which is not CFL, but a subset of finite set has to be finite.

For example.

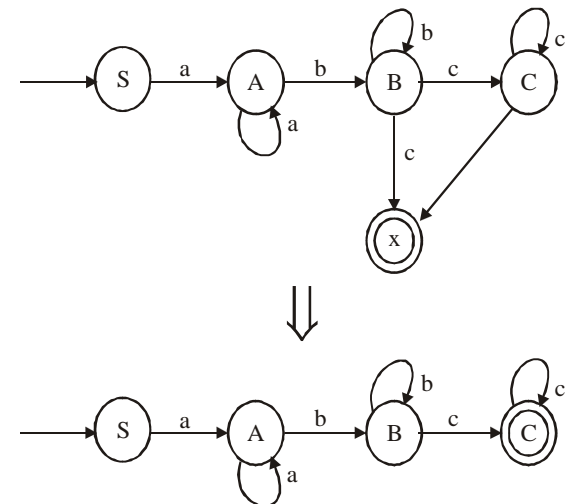
1.  $L = (0 + 1)^*$  is regular. Let  $L_s = \{0^n 1^n\}$

Here  $L_s \subseteq L$ . But  $L_s$  is not regular.

2.  $L = (0 + 1)^*$  is regular and hence also CFL. Consider

$L_s = \{0^n 1^n 0^n | n \geq 0\}$  Here  $L_s \subseteq L$ , but  $L_s$  is not a CFL.

52. (c) The given grammar is right linear.



53. (c)  $L_1 = a^m b^n | n = m^2$

$$L_2 = a^m b^m c^m d^n | m, n > 0$$

$$L_3 = a^m b^m c^n d^n | m, n > 0$$

In  $L_1$ ,  $n = m^2$  cannot be checks in stack as  $m^2$  is a nonlinear function. Hence it is not CFL.



$L_2$  requires 2 comparisons on no. as itself.

$\Rightarrow$  no. of a's = no. of b's and no. of c's. This cannot be done by single stack.

$L_3$  requires 2 comparisons, but only one at a time is required

$\Rightarrow$  no. of a's = no. of b's and no. of c's = no. of d's. This cannot be done by single stack.

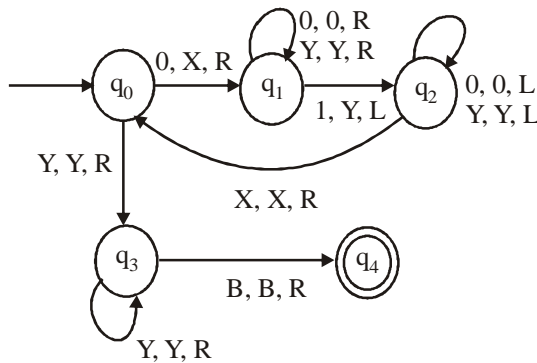
54. (c)  $L_1$  and  $L_2$  are context free languages. We can see that  $L_2$  is in fact a regular language having regular expression  $a(ba)^*$ .

$L_1$  is a language containing strings having equal number of a's and b's

$\Rightarrow L_3 = L_1 \cap L_2$  is a context free language.

$\therefore$  class of context free languages is closed under concatenation and kleene closure. Therefore  $L_4 = L_1 \cdot L_1^*$  is a context free.

55. (c) Drawing the equivalent turing machine we have



56. (d)  $R_1 = (a | b)^* = (a + b)^* \dots(1)$   
 $R_2 = (a^* | b^*)^* = (a^* + b^*)^* = (a + b)^* \dots(2)$   
 $R_3 = (\epsilon | a | b^*)^* = (\epsilon + a + b)^* = (a + b)^* \dots(3)$

Clearly from (1), (2) and (3)

57. (b) Clearly, we can observe that the minimum string accepted is aabb.

The corresponding  $L(M) = \{W/W \in (a + b)^* aabb (a + b)^*\}$

58. (b) The correct statement follows from following theorem:  
**Theorem:** If  $L_1$  and  $L_2$  is a pair of complementary language then either

- Both  $L_1$  and  $L_2$  are recursive.
- Neither  $L_1$  and  $L_2$  are recursively enumerable.
- One is recursively enumerable but not recursive, the other is not RE.

Option (b) is not possible, since if  $L_1$  is recursive, then

$L_2 = L_1^c$  will also be recursive.

59. (c) B and C are useless, since they are not generating any terminal.

Therefore the remaining productions left after removing useless, unit and  $\lambda$ -productions

$S \longrightarrow aA \mid a, A \longrightarrow aa \mid aaA$

60. (a) The TM carries out the functions of  $(m - n)$  in  $(0^m 10^n)$   
 $\rightarrow$  when  $m \geq n$ , the output is difference between  $m$  and  $n$ .  
 $\rightarrow$  when  $m < n$ , the output is blank.

61. (c)

62. (b) Suppose,  $S = aaab, |s| = 4$ . The prefixes are  
 $S_p = \{\lambda, a, aa, aaa, aaab\}$ . Here  $aaab$  is not a proper prefix.

**Note:** The proper prefix of string  $S$  is a prefix, which is not same as string  $S$ .

A string of length 4 has 4 proper prefixes. A string of length 5 has 5 proper prefixes. For a string of length  $n$ , therefore we can have ' $n$ ' proper prefixes.

63. (a) The following CFG

$S \rightarrow aB \mid bA$

$A \rightarrow ba \mid aS \mid bAA$

$B \rightarrow b \mid bS \mid aBB$

by observing the given CFG we can see that with each non-terminal, number of a's and b's get equalled.

The language is the strings of terminals that have equal number of a's and b's

64. (a) Rule:  $a^* (a^*b)^* = a^* (ba^*)^* \neq (a + b)^*$

So,  $0^* (10^*)^* = 1^* (01^*)^*$

65. (c) The strings with the first, 4th and 7th bits as '1' will look in the following format

1 \_ \_ 1 \_ \_ 1.

So, there can be 16 possible combinations for the above format. But in the given DFA, only 7 strings of these will be accepted. They are,

1. 1001001
2. 1001011
3. 1001101
4. 1001111
5. 1011001
6. 1101001
7. 111001

66. (b) Diagonalization language,  $L_d$  is defined as

$L_d = \{TM \text{ that accepts its own code}\}$

$L_d$  is clearly RE but not REC.

67. (c) Single tape turing machine can simulate  $n$ -moves of  $k$ -tape TM in  $O(n^2)$

68. (b) Since  $0^* (10^*) = 0^* (1^*0^*)^*$  for

(a)  $(1^*0)1^* = (1^*0^*)1^*$

286

- (b)  $0 + (0 + 10)^* = 0 + (0^* 1^* 0^*) = (0^* 1^* 0^*)$   
 (c)  $(0 + 1)^* 10(0 + 1)^*$

69. (b)  $L \in NP$  does not imply that  $L' \in NP$ . Since, closure of NP under complementation is as yet unresolved problem. Also  $L \in NP$  does not imply  $L' \in P$ , unless  $L \in P$ , which may or may not be so.

70. (c) P, Q and R be three languages

Let  $P = (a + b)^*$

$Q = ab$

Then  $PQ = (a + b)^* ab = R$

Here P, Q and R are regular.

Let  $P = \phi$

$Q = \{a^n b^n \mid n \geq 0\}$

$PQ = R = \phi$

Here P and R are regular but Q is not

So Q need not to be regular

71. (b) Proof: First, we have to prove that TSP belongs to NP. If we want to check a tour for credibility, we check that the tour contains each vertex once. Then we sum the total cost of the edges and finally we check if the cost is minimum. This can be completed in polynomial time thus TSP belongs to NP. Secondly we prove that TSP is NP-hard. One way to prove this is to show that Hamiltonian cycle TSP (given that the Hamiltonian cycle problem is NP-complete). Assume  $G = (V, E)$  to be an instance of Hamiltonian cycle. An instance of TSP is then constructed. We create the complete graph  $= (V, P \leq G'E')$ , where  $E' = \{(i, j) : i, j \in V \text{ and } i \neq j\}$ . Thus, the cost function is defined as:

$t(i, j) = 10.1 \notin j \mid (i, j) \in E, \text{ if } 1, j \mid (i, j) \notin E$

Now suppose that a Hamiltonian cycle h exists in G. It is clear that the cost of each edge in h is 0 in G as each edge belongs to E. Therefore, h has a cost of 0 in G'. Thus, if graph G has a Hamiltonian cycle then graph G has a tour of 0 cost.

Conversely, we assume that G' has a tour h' of cost at most 0. The cost of edges in E' are 0 and 1 by definition. So each edge must have a cost of 0 as the cost of h' is 0. We conclude that h' contains only edges in E.

So we have proven that G has a Hamiltonian cycle if and only if G' has a tour of cost at most 0.

Thus TSP is NP-complete.

72. (d) Euclidean algorithm for finding gcd(x, y):

function gcd(hx, yi)

if (y = 0) return x else return gcd(hy, x mod yi)

Theorem

The Euclidean algorithm called on input hx, yi runs in time

$O(\log(x + y))$ .

Hence its running time is  $O(n)$  (its input is encoded in binary).

Conclusion

$RELPRIME \notin P$

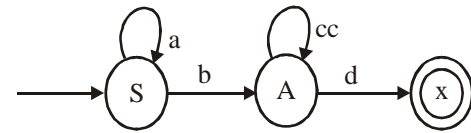
73. (c) If there is an NP-complete language whose complement is in NP. Then the set  $NP = CO-NP$  which implies  $NP = P$ .

i.e.  $NP = CO-NP = P$

So complements of any language is NP will in NP as well P.

So, both (a) & (b) are true choice (c) is correct.

74. (c)  $S \rightarrow aS/bA, A \rightarrow d/ccA$



We can clearly see that X is a final state. And the corresponding language regular expression is  $a^*b(cc)^*d$

75. (d) Let start with S and develop some string

1.  $S \rightarrow PQ$  using  $S \rightarrow PQ$

2.  $S \rightarrow xQ$  using  $P \rightarrow x$

3.  $S \rightarrow xy$  using  $Q \rightarrow y$

Hence  $n = 2$  and number of productions required is 3

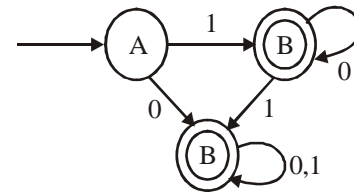
Now check  $n^2$  i.e.  $2^2 \neq 3$  eliminates (a)

$2n = 2.2 = 4 \neq 3 \therefore$  eliminates (a)

$2n + 1 = 2.2 + 1 = 8 \neq 3 \therefore$  eliminates (b)

$2n - 1 = 2.2 - 1 = 3 = 3$

76. (a) For (a) we can construct a DFA



which will recognize the series

$1, 2, 4, \dots, 2^n, \dots$  in binary as

$1, 10, 100, 1000, \dots$

Choice (b) is  $\{1^{2^n} ; n \geq 0\}$

Choice (c) is  $\{w \mid n_0(w) = n_3(w)\}$

Choice (d) is  $\{1^n 01^n, n \geq 0\}$

DFA's cannot be constructed for b, c and d since these are not regular languages.



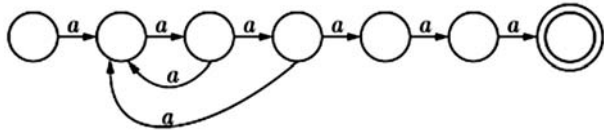
77. (d) The required regular expression is

$$r = b^+ + ab^+ + \epsilon$$

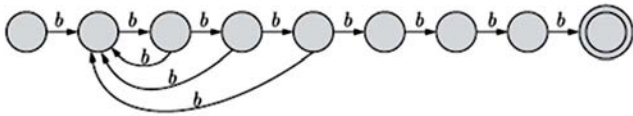
$$= (\epsilon + a)b^+ + \epsilon$$

$$r = b^* + ab^+$$

78. (c) In *DFA* the no. of states are always more than *NFA*, so if *NFA* has  $N$  states *DFA* will have  $2N$  states.  
79. (c) The valid strings will be where no. of  $a$ 's 6, 12, 18, 24  
No. of  $b$ 's = 8, 16, 24



No. of states = 7 for  $a$



No. of states = 9

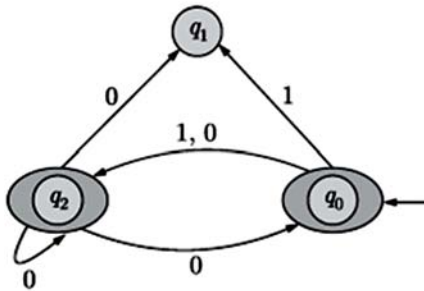
Total  $9+7-1 = 15$

1 subtracted due to 2 final states.

80. (c)  $L_1$  would be accepted by *PDA* so can't be regular.  
 $L_2$  similarly can't be accepted by *DFA* so not regular.  
 $L_3$  &  $L_4$  both require only finite no of zeros.  
So both regular.
81. (a) Since it is possible to create a turing machine for the problem, 20 this problem is decidable.
82. (b) Pushdown Automaton uses stack as data structure & languages accepted by *PDA* is regular.
83. (c) A problem is said to be *NP*-complete, if it is both *NP* & *NP* hard. 3-SAT problem is *NP* complete so a reduction of 3-SAT problem to  $\Pi$  &  $\Pi$  to 30-SAT.  
So this infers that  $\Pi$  is *NP* complete, since it is reducible to a *NP* complete problem.
84. (a) A language  $L$  is said to be recursive if there exists any rule to determine whether an element belong to language or not, if language can be accepted by turning machine.  
So there exist the rules so  $L$  is recursive.
85. (b) Since  $L$  can be effectively enumerated so  $L$  has to be regular, but is doesn't mean that the decisions are finite.
86. (c) The strings accepted by the given automata are of type.  
Option  
1 2 3 4 5 6 7  
1--1--1  
These four blank spaces can have 0 or 1, so total  $2_4 = 6$  strings are possible, but the given automata does not accept all of those.  
1. 1 1 1 1 0 0 1  
2. 1 1 0 1 0 0 1  
3. 1 0 1 1 0 0 1  
4. 1 0 0 1 0 0 1  
5. 1 0 0 1 0 0 1  
6. 1 0 0 1 1 0 1  
7. 1 0 0 1 1 1 1
87. (c) (A) Incorrect since the production has same non terminal in both sides, so definitely ambiguous.  
(B) Since  $S''SS$  this leads to conjunction of every possible string to make a valid string in  $L(G)$ .  
(C) Context free languages are accepted by push down automata so true.  
(D) The language is not regular so *DFA* is not possible.
88. (a)  $S_1$  can be represented using a *DFA* so it is regular  $S_1$  is correct.  
 $S_2$  can't be represented by *DFA* but it requires *PDA* to accept. So is  $S_2$  is *CFG* not regular.  $S_2$  is false.
89. (b) (A) It is not necessary at all.  
(B)  $\{a^n b^n\} \cup \{a^n b^n c^n\} = \{a^n b^n c^n\}$  always true so correct.  
(C)  $\{a^n b^n\} \cap \{a^n\} = \{a^n\}$  not *CFG* so false.  
(D) Not necessary.
90. (b)
- 
- Start & end are same (a) so the minimum no. of states required are 3.  
If string traversal doesn't stop at (a) then string length is not divisible by 3.
91. (a)  $C$  language is context free language entirely based upon the productions.
92. (b) Pushdown Automaton uses stack as data structure & languages accepted by *PDA* is regular.
93. (c) A problem is said to be *NP*-complete, if it is both *NP* & *NP* hard.  
3-SAT problem is *NP* complete so a reduction of 3-SAT problem to  $\Pi$  &  $\Pi$  to 30-SAT.  
So this infers that  $\Pi$  is *NP* complete, since it is reducible to a *NP* complete problem.
94. (a) A language  $L$  is said to be recursive if there exists any rule to determine whether an element belong to language or not, if language can be accepted by turning machine.  
So there exist the rules so  $L$  is recursive.
95. (b) Since  $L$  can be effectively enumerated so  $L$  has to be regular, but is doesn't mean that the decisions are finite.
96. (c) The strings accepted by the given automata are of type.  
Option  
1 2 3 4 5 6 7  
1--1--1  
These four blank spaces can have 0 or 1, so total  $2_4 = 6$  strings are possible, but the given automata does not accept all of those.  
1. 1 1 1 1 0 0 1  
2. 1 1 0 1 0 0 1  
3. 1 0 1 1 0 0 1

4. 1 0 0 1 0 0 1
5. 1 0 0 1 0 0 1
6. 1 0 0 1 1 0 1
7. 1 0 0 1 1 1 1

97. (A) Incorrect since the production has same non terminal in both sides, so definitely ambiguous.  
 (B) Since  $S \rightarrow SS$  this leads to conjunction of every possible string to make a valid string in  $L(G)$ .  
 (C) Context free languages are accepted by push down automata so true.
98. (a) This turning machine starts at 90 if it doesn't get any input symbol but  $B$  then it halts.  
 So if  $(00+1)^*$  is chosen then the  $M/C$  can halt. Option (b) is wrong.  
 Option (c) & (d) are possible but not necessary.  
 Option (a)  $(0+1)^*$  1 or more occurrence of 0 or 1.  
 So 0, 1, 00, 01, 10, 11.....are valid strings & the machine doesn't halt for them.
99. (a)  $L$  is accepted by M(NFA) but NFA  $M_1$  has



So this accept  $L_1$ .  
 $L_1$  will accept not only  $L$  but also substrings of  $L$ .  
 So  $L_1$  is a subset of  $L$

100. (c) 3 SAT problem is both NP & NP hard so it is NP complete, but 2 SAT problem is solvable in Polynomial time so it is in class P.
101. (a) Due to the 3 one's in the upper edges & 3 one's in lower edges to reach to final state the no of 1's is always divisible by 3 & 0's are always in pair in forward & back edge so, no of zero's is divisible by 2.
102. (b) Language  $\{a^n b^n c^{m+n} / m, n \geq 1\}$  is a context free language since it can be represented by pushdown automata, but it is not regular since  $\Delta FA$  can't count the no. of  $a$ 's &  $b$ 's and then check the sum for occurrence of  $c$ .
103. (c)  $S \rightarrow bS / aA / b$   
 $A \rightarrow bA / aB$   
 $B \rightarrow bB / aS / a$   
 Let  $N_a(w)$  &  $N_b(w)$  denote of  $a$ 's &  $b$ 's in strings.  
 Some valid strings are  
 1.  $S \rightarrow bS \rightarrow bbS \rightarrow bbb$  (any no. of  $b$ )

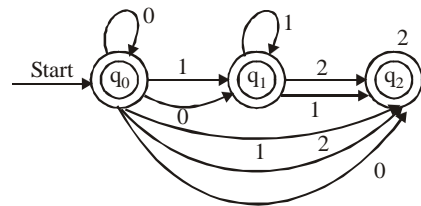
2.  $S \rightarrow bA \rightarrow abA \rightarrow abbaA \rightarrow abbaB \rightarrow abbaa$
3.  $abbaB \rightarrow abbaaS \rightarrow abbaaab$

From (2) option (D) is false also from (1), (2) & (3) (a), (b) & (d) are false.

So only (C) satisfy

104. (c)  $P_1 \rightarrow$  decidable  
 $P_2 \rightarrow$  undecidable  
 If  $P_1$  or  $P_2$  is reducible to  $P_3$  then  $P_3$  also has same properties as  $P_1$  &  $P_2$ .  
 So if  $P_2$  is reducible to  $P_3$  then  $P_3$  is also undecidable.
105. (b) From the given FSM, it is clear that a not necessity followed by only  $2b$  due to self loop at final state. But at least  $2b$ 's are there.  $abb$  substring not always, Similarly  $aa$  not always.
106. (b) The rules here used will be.  
 All those languages which are recursive their complements are also recursive.  
 So option (a) & (b) can be correct.  
 Now languages which are recursively enumerable but not recursive, their complements can't be recursively enumerable. So only option (b) is correct
107. (b) In all the options there is linear relationship among strings so all CFL's, but  $L_1$  &  $L_3$  can be accepted by PDA,  $L_2$  can be accepted by deterministic CFL due to presence of special symbol  $D$  which tells the middle of the string, so deterministic.

108. (c) The given NFA when converted to NFA without  $\epsilon$ -moves.



**Note:** The previous state from which there is an  $\epsilon$ -Move, if it goes to the final state then it also becomes the final state as well.

109. (c) From the above NFA we can clearly see that it accepts strings only if "0 followed by 1 followed by 2" i.e. strings like 2 followed by 0 or b and 1 followed by - are not accepted. Clearly, (c) is the correct option.
110. (b) 111. (c) 112. (b)
113. (c) Since both A and B are nullable therefore S is also nullable. Therefore all A, B and S are nullable.

114. (d)

$$\begin{aligned} 115. (b) \quad \text{Val}(a, (a + a^*aa)) &= \max(\text{val}(a), \text{val}(a + a^*aa)) \\ &= \max(0, \text{val}(a + a^*aa)) \end{aligned}$$

$$\begin{aligned} \text{val}(a) &= 0 \\ &= \text{val}(a + a^*aa) \\ &= \max(\text{val}(a), \text{val}(a^*aa)) \\ &= \max(0, \text{val}(a^*aa)) \\ &= \text{val}(a^*aa) \\ &= \max(\text{val}(a^*), \text{val}(aa)) \\ &= \max(\text{val}(a^*), 0) \\ &= \text{val}(a^*) \\ &= \text{val}(a) + 1 = 0 + 1 \\ &= 1 \end{aligned}$$

$$\begin{aligned} 116. (b) \quad \text{val}((a(a + a^*aa) + aaa)^*) &= \text{val}(a(a + a^*aa) + aaa)^* \\ &= \max(\text{val}(a^*a + a^*aa), \\ &\quad \text{val}((aaa)) + 1) \\ &= \max(1, 0) + 1 \\ &= 1 + 1 = 2 \end{aligned}$$

$$\begin{aligned} 117. (a) \quad \text{Val}(a^*b^*b^*b^*)^* &= \text{val}(a^*b^*b^*b^*) + 1 \\ &= \text{val}(a^* + b^* + b^* + b^*) + 1 \\ &= \max(\text{val}(a^*), \text{val}(b^*)) + 1 \\ &= \max(\text{val}(a) + 1, \text{val}(b) + 1) + 1 \\ &= \max(1, 1) + 1 \\ &= 1 + 1 = 2 \end{aligned}$$