**PART B**

1. **Consider G1={{S},{a, b}, P1, S} where P1= S->aSb/ab .G2={{S,A,B,C},{a, b}, P2, S} where P2={S->AC, S->AB, A->a, C->SB, B->b}**
   1. **Show that G1 is equivalent to G2.**
   2. **Construct the derivation tree for the string aaaabbbb using G1 and also with G2.**

**Solution:**

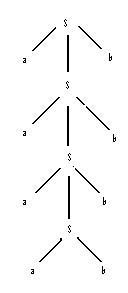
* + - 1. Language accepted by G1= {anbn}

Language accepted by G2= {ambm}

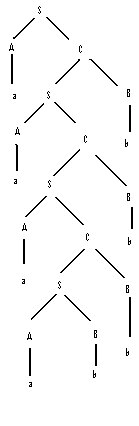
L(G1) =L(G2)

Hence G1 is equivalent to G2.

* + - 1. Using G1



Using G2



1. **For the DFA, M={Q, ∑, ∂, F, S}, Q= {q0, q1, q2, q3}, ∑= {0, 1}, S = q0, F = q0 and the transition table is given as follows**

|  |  |  |
| --- | --- | --- |
| **States** | **0** | **1** |
| **q0** | **q2** | **q1** |
| **q1** | **q3** | **q0** |
| **q2** | **q0** | **q3** |
| **q3** | **q1** | **q2** |

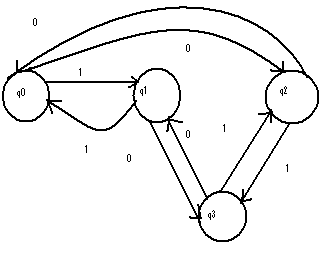
1. **Draw the transition diagram for the DFA**
2. **Check whether the following strings are accepted by the DFA**
   1. **110101 b. 001110 c. 101000**

**Solution:**

In a Deterministic Finite State Automata there is a unique next state for every pair

of state and input.

1)



2) 110101 Accepted

001110 Not Accepted

101000 Accepted

1. **An NFA with states {1, 2, 3, 4, 5}, ∑ = {a, b} has the following transition table:**

|  |  |  |
| --- | --- | --- |
| **States** | **Input**  **a b** | |
| **1** | **{1,2}** | **{1}** |
| **2** | **{3}** | **{3}** |
| **3** | **{4}** | **{4}** |
| **4** | **{5}** | **Ф** |
| **5** | **Ф** | **{5}** |

1. **Calculate ∂(1, ab)**
2. **Calculate ∂(1, abab)**
3. **Calculate ∂(1, abba)**
4. **Calculate ∂(1, abaa)**

**Solution:**

1) ∂(1,ab)={1,3}

2) ∂(1,abab)={1,3}

3) ∂(1,abba)={1,2,5}

4) ∂(1,abaa)={1,2,3,5}

1. **Given the following NFA?**

**M = {{q0, q1, q2}, {a, b}, ∂, q0, q2} and**

|  |  |  |
| --- | --- | --- |
|  | **A** | **b** |
| **q0** | **{q0,q1}** | **{q2}** |
| **q1** | **{q0}** | **{q1}** |
| **q2** | **Ф** | **{q0,q1}** |

1. **For the corresponding DFA find Q’**

**2) Find ∂' for each state with all possible inputs**

**Solution:**

1. Q'={[q0], [q1], [q2], [q0,q1], [q0,q2], [q1,q2], [q0,q1,q2]}

2) ∂' for each state with all possible inputs

|  |  |  |
| --- | --- | --- |
|  | a | b |
| [q0] | [q0,q1] | [q2] |
| [q1] | [q0] | [q1] |
| [q2] | Ф | [q0,q1] |
| [q0,q1] | [q0,q1] | [q1,q2] |
| [q0,q2] | [q0,q1] | [q0,q1,q2] |
| [q1,q2] | [q0] | [q0,q1] |
| [q0,q1,q2] | [q0,q1] | [q0,q1,q2] |

1. **Examine whether the following grammar is ambiguous or not.**

**G = {N, T, S, P}, where N = {S, A}, T = {a, b}, P = {S→aAb, S→abSb, S→a, A→bS, A→aAAb}**

**Solution:**

S⇒aAb

⇒abSb using A→bS

⇒abab using S→a

S⇒abSb

⇒abab using S→a

Thus abab is generated by two leftmost derivations. The grammar G is ambiguous.

1. **Construct a grammar generating all palindromes over {a, b}.**

**Solution:**

Define G = {{S}, {a, b}, P, S} where P = {S→aSa, S→bSb, S→a, S→b, S→λ}

S⇒ aSa⇒ aaSaa⇒ aabSaab⇒aabbaa etc

Thus a palindrome of even length can be generated by S→ aSa, S→ bSb and S→λ

Similarly, a palindrome of odd length with centre a can be generated by using

S→aSa/bSb/a and a palindrome of odd length with centre b can be generated by using

S→aSa/bSb/b. Terefore G is the required grammar.

1. **Consider the NFA defined by the state diagram given below. Convert it to a DFA.**

b

a

b

b

**Solution:**

The states of the DFA are {S0},{S1}, {S1, S2}and Φ.

The state diagram of the equivalent DFA is

a

a

b

a

b

a

b

b

1. **Draw the state diagram of the FSA, the accepting states of which are S0 and S3 and for which the state table is as shown below in the table. Find also the language accepted by this FSA.**

|  |  |  |
| --- | --- | --- |
|  | **f** | |
| **S \ I** | **0** | **1** |
| **S0** | **S0** | **S1** |
| **S1** | **S3** | **S2** |
| **S2** | **S2** | **S2** |
| **S3** | **S3** | **S3** |

**Solution:** The state diagram is

0, 1

0

1

1

0

0, 1

The strings that take the FSA from S0 to S0 are acceptable by the FSA. Such strings can have any number of zeros. i.e., 0n, n≥0. The strings that take the FSA from S0  to S3 are also acceptable by the FSA. Such strings can have any number of 0’s followed by 10. i.e., 0n10, n≥0. Any number of 0’s followed by 10 and then followed by any other string x will also be accepted by the FSA. Hence the language generated and accepted by the FSA is L = {0n, 0n10, 0n10x / n = 0, 1, 2, 3, …}, where x is any string.

1. **Construct a finite state machine accepting exactly the strings with an even number of 1’s**

**Solution:**

As our concern is only the number of 1’s, the automation can ignore 0’s by not changing state and giving an output 0. On seeing a 1, it can change state with output 0. On seeing a second 1, it can go back to the starting state with output 1.

The required FSM is defined as M = {{S0, S1}, {0, 1}, {0, 1}, f, g}

The state transition function f and the output function g are defined by the state table.

|  |  |  |
| --- | --- | --- |
| **f, g** | **0** | **1** |
| **S0** | **S0, 0** | **S1, 0** |
| **S1** | **S1, 0** | **S0, 1** |

The state diagram defining f and g is

0/0

0/0

1/0

1/1

1. **Find a grammar G such that L(G) = {anbn / n**≥**1}**

**Solution:**

Define G = {{S}, {a, b}, P, S} where P = {S→aSb, S→ab}

Then S⇒ab and therefore ab ε L(G).

S⇒ aSb⇒ aaSbb⇒ … ⇒ an-1Sbn-1⇒ an-1abbn-1= anbn

Therefore anbn ε L(G).

Conversely, let w ε L(G). Assume S⇒w in n steps.

If n = 1, and use S→ab once to get w = ab.

If n ≠ 1, and S→aSb is used in the first n-1 steps and S→ab is used in the nth step,

then w = anbn . Hence L(G) = {anbn/ n≥1}

Thus G= {{S}, {a, b}, P, S} where P = {S→aSb, S→ab}

**11Construct a grammar for the language L = {ambn / m>n>0}**

**Solution:**

Define G = {{S, A}, {a, b}, P, S} where P = {S→aS, S→aA, A→aAb, A→ab}

ambn = am-nanbn

Now S⇒ am-n-1S by applying S→ aS m-m-1 times

⇒ am-n-1aA = am-nA by applying S→ aA

⇒ am-nan-1Abn-1 by applying A →aAb n-1 times

⇒ am-1abbn-1 = ambn by applying A→ ab

Hence ambn ε L(G) when m>n>0

Conversely, let w ε L(G)

Then S ⇒ akS if S→ aS is used in k steps

⇒ akaA if S→ aA is used

⇒ ak+1aiAbi if A→ aAb is used in i steps

⇒ ak+i+1abbi if A→ ab is used

⇒ ambn where m>n>0

S ⇒ aA if S→ aS is used in k steps

⇒ aaiAbi if A→ aAb is used in i steps

⇒ ai+1abbi if A→ ab is used

⇒ ambn where m>n>0

Therefore, w = ambn for m>n>0

Hence L(G) = {ambn/ m> n>0}

Thus G= {{S, A}, {a, b}, P, S} where P = {S→aS, S→aA, A→aAb, A→ab}

1. **The state table of a finite state machine M is given by**

|  |  |  |
| --- | --- | --- |
| **f, g** | **a** | **b** |
| **S0** | **S0, b** | **S4, b** |
| **S1** | **S0, a** | **S3, b** |
| **S2** | **S0, a** | **S2, a** |
| **S3** | **S1, b** | **S1, b** |
| **S4** | **S1, b** | **S0, a** |

1. **Find the input set I, the state set S, the output set O and the initial state of M**
2. **Draw the state diagram of M**
3. **Find the output of the word w = a2bab2a**

**Solution:**

1. I = {a, b} S = {S0, S1, S2, S3, S4} The initial state is S0.
2. The state diagram is

a, b

b, b

b,a

a, b

b, b

a, a

a, a

a, b

b,a

b, b

1. The sequential arrow diagram giving the output of the given word is



Hence the output is b6a.

1. **The state table of a finite state machine M is given in the table below.**
2. **Identify the input set I, the state set S, the output set O and the initial state of M**
3. **Draw the state diagram of M**
4. **Find the output of the string 0212012021**

|  |  |  |
| --- | --- | --- |
| **f, g** | **0** | **1** |
| **S0** | **S2, y** | **S1, z** |
| **S1** | **S2, x** | **S3, y** |
| **S2** | **S2, y** | **S1, z** |
| **S3** | **S3, z** | **S0, x** |

**Solution:**

1. I = {0, 1} S = {S0, S1, S2, S3} O = {x, y, z} Initial state is S0
2. The state diagram of M is

1, y

1, z

0, z

0, x

0, y

1, z

0, y

1, x

1. The sequential arrow diagram which gives the output for the string 0212012021 is



The output is the string y2zyzxzxyz

1. **Design an FSM that performs serial binary addition.**

**Solution:**

The serial adder requires input set I = {00, 01, 10, 11} and output set O = {0, 1}. If the input pair is ab, we add a and b if the carry bit is 0 and add a, b and 1 if the carry bit is 1. Hence the states of the machine are S0 representing the absence of a carry and S1 representing the presence of e carry. The transition function f is defined as follows.

f(S0, 11) = S1 f(S0, 00) = S0  f(S0, 01) = S0

f(S0, 10) = S0 f(S1, 11) = S1 f(S1, 00) = S0

f(S1, 01) = S1 f(S1, 10) = S1

The state table is

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **f, g** | **00** | **01** | **10** | **11** |
| **S0** | **S0, 0** | **S0, 1** | **S0, 1** | **S1, 0** |
| **S1** | **S0, 1** | **S1, 0** | **S1, 0** | **S1, 1** |

11,1

10,0

01,0

01,1

10,1

00,1

a

00,0

11,0