MATHEMATICAL FOUNDATION FOR COMPUTER SCIENCE

Prepared by: Er. Ankit Kharel

Nepal college of information technology

FINITE STATE AUTOMATA

- *Sequential Circuits and Finite state Machine*
- *Finite State Automata*
- *Non-deterministic Finite State Automata*
- *Language and Grammars*
- *Language and Automata*
- *Regular Expression*

REGULAR EXPRESSIONS:

RG
(Regular Grammar)

A Language is regular if there is a regular grammar to generate it.

RL
(Regular Language)

FA
(Finite Automata)

A Language is regular if there is a exist a finite automata to accept it.

RE
(Regular Expression)

A Language is regular if there is a exist a regular expression to express it.

REGULAR EXPRESSIONS:

The language accepted by finite automata can be easily described by simple **expressions** called **Regular Expressions**. A **regular expression** can also be described as a sequence of pattern that defines a string. **Regular expressions** are used to match character combinations in strings.

For instance:

In a regular expression, x^* means zero or more occurrence of x. L(R) = {e, x, xx, xxx, xxxx, ...}

In a regular expression, x^+ means one or more occurrence of x. L(R) = {x, xx, xxx, xxxx, ...}

Let 'R' be a regular expression over alphabet Σ:

a) ε is a Regular Expression denoting the set:

R= ε ; L(R) = {ε}

b) ϕ is a Regular Expression denoting the empty set:

R= ϕ ; L(R) = { }

c)For each symbol a$\in \Sigma$, a is regular expression denoting set {a}.

R=a ; L(R) = {a}

a, b, and c are called primitive regular expression. It is the minimum language generated by RE.

REGULAR EXPRESSIONS:

(Operators used in Regular expression)

d. **Union(+)** of two RE is also Regular;

$R_1 = a$, $R_2 = b$, $R_1 \cup R_2 = a + b$ i.e. $R_1 \cup R_2 = a + b$ generates language that contains either a or b

e. **Concatenation(.)** of two RE is also Regular;

$R_1 = a$, $R_2 = b$, $R_1.R_2 = a.b$ i.e. $R_1.R_2 = a.b$ generates language that contains a and b.

f. **Kleene Closure** of RE is also regular;

$R_1 = a$, $R_1^* = a^*$

g. **Positive Closure** of RE is also regular;

$R_1 = a$, $R_1^+ = a^+$

REGULAR EXPRESSIONS:

Q. Find the regular expression for the following languages,

1. Language containing no string:

   R = ϕ

2. Language containing string of length 0:

   R = ε

3. Language accepting string of length 1 over $\Sigma = \{a, b\}$.

   L = (a, b)

   R = a+b

4. Language accepting string of length 2 over $\Sigma = \{a, b\}$..

   L= (aa , ab , ba , bb)

   R = aa + ab + ba + bb

   = a(a+b) +b(a+b)

   =(a+b)(a+b)

REGULAR EXPRESSIONS:

5. Language accepting any combination of a over $\Sigma = \{a\}$

   R = a^*

6. Language accepting any combination of a expect null over $\Sigma = \{a\}$

   R = a^+

7. Language accepting all the string containing any number of a's and b's over $\Sigma = \{a, b\}$.

   R = $(a + b)^*$

8. Language accepting string of length at most 2 over $\Sigma = \{a, b\}$..

   L= (ε , a, b, aa , ab , ba , bb)

   R = ε + a + b + aa + ab + ba + bb

REGULAR EXPRESSIONS:

9. Language accepting all string having a single b over $\Sigma = \{a, b\}$.

   R = a^*ba^*

10. Language accepting all string having at least one b over $\Sigma = \{a, b\}$.

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

11. Language accepting all the string containing any number of a's and b's over $\Sigma = \{a, b\}$.

    R = $(a + b)^*$

12. Language containing string with ‘bbbb’ as substring over $\Sigma = \{a, b\}$.

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

13. Language containing string that ends with ‘ab’ over $\Sigma = \{a, b\}$.

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

14. Language containing string that starts with ‘ab’ over $\Sigma = \{a, b\}$.

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

REGULAR EXPRESSIONS:

15. Language accepting all string that starts with a and ends with a over $\Sigma = \{a, b\}$.
    R = a + a(a+b)*a

16. Language accepting all string that starts and ends with same symbol over $\Sigma = \{a, b\}$
    R = a(a+b)*a + b(a+b)*b + a + b

17. Language accepting all string that starts with a and ends with b over $\Sigma = \{a, b\}$.
    R = a(a+b)*b

18. Language accepting all string that starts and ends with different symbol over $\Sigma = \{a, b\}$
    R = a(a+b)*b + b(a+b)*a

19. Language accepting string that contains exactly two b's over $\Sigma = \{a, b\}$.
    R = a* b a* b a*

20. Language containing string that starts with 'ab' over $\Sigma = \{a, b\}$.
    R = ab (a+b)*

REGULAR EXPRESSIONS:

21. Language accepting all string where number of a is less than or equal to 2 over $\Sigma = \{a, b\}$.

    R = $b^* + b^*a\, b^* + b^*a\, b^*a\, b^*$

22. Language accepting all string where 3rd symbol from LHS is b over $\Sigma = \{a, b\}$

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

    $= (a+b)^2 b(a+b)^*$

23. Language accepting all string where every 0 is followed by immediate 11over $\Sigma = \{0, 1\}$.

    R = $(1 + 011)^*$

24. Language accepting all string where 2nd symbol from RHS is b over $\Sigma = \{a, b\}$

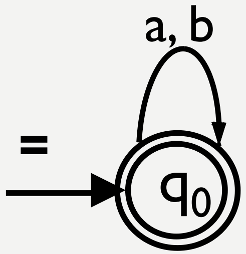
    R = $(a+b)^*b(a+b)$

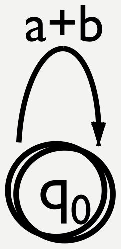
25. Second symbol is a and fourth symbols is b.

    R = $(a+b)a(a+b)b\,(a+b)^*$

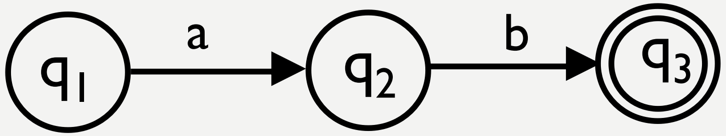
CONVERSION OF FINITE AUTOMATA TO RE:

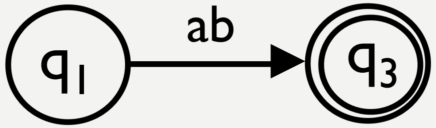
State Elimination Method:

1. = 

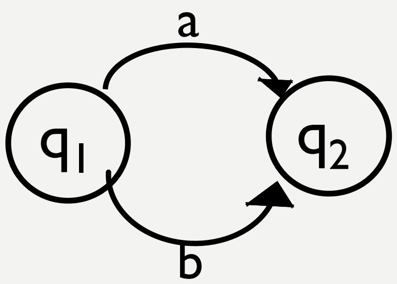
= 

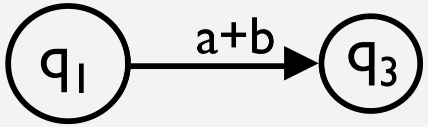
= $(a + b)^*$

2. = 

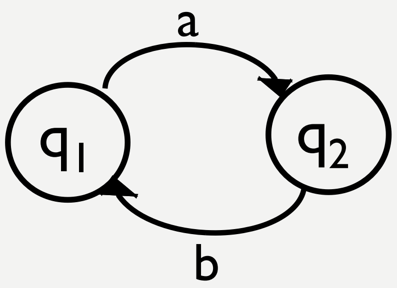
= 

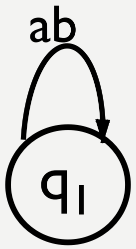
= a.b

3. = 

= 

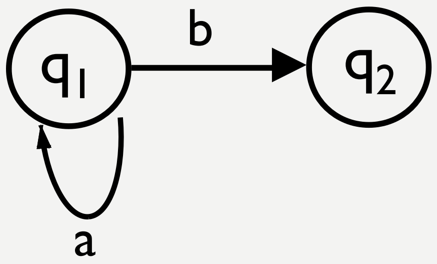
= (a+b)

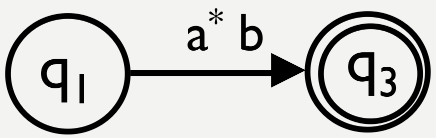
4. = 

= 

= $(ab)^*$

CONVERSION OF FINITE AUTOMATA TO RE:

4= 

= 

= $\mathbf{a^*.b}$

Steps to convert FA to RE:

1.If there exists any incoming edge to the initial state , create a new initial state having no incoming edge.

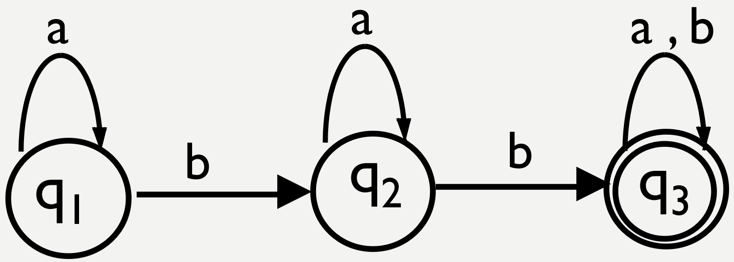
2. In case of multiple final states , convert them into non final state and create a new final state.

3.If there exist outgoing edge from final state create new final state having no outgoing edge.

4. Eliminate all intermediate state one by one . Only initial and Final state will be there. Their transition path will be our R.E.

CONVERSION OF FINITE AUTOMATA TO RE:

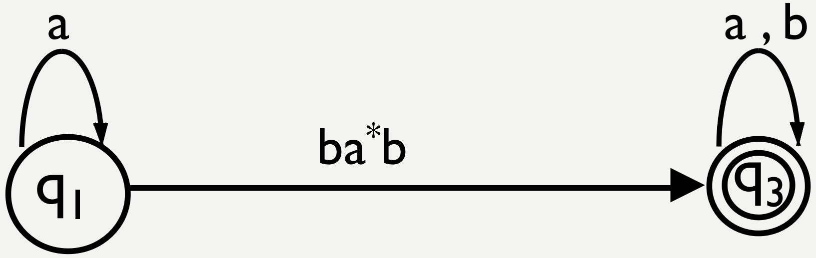
1. Convert Following to R.E



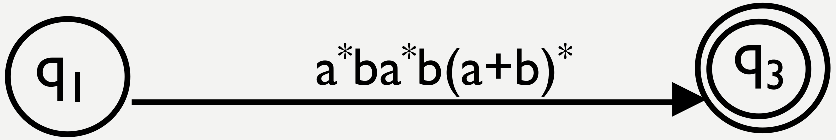
Solution:

i)Since there is no incoming edge in initial state, no outgoing edge from final state and there exist only one final state. So, start removing intermediate state .

=



=

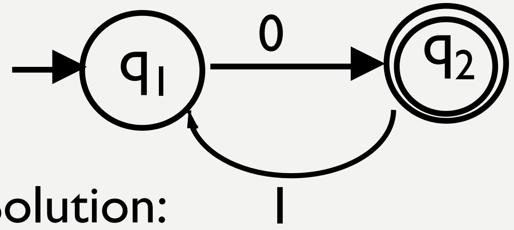


The Required R.E is:

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

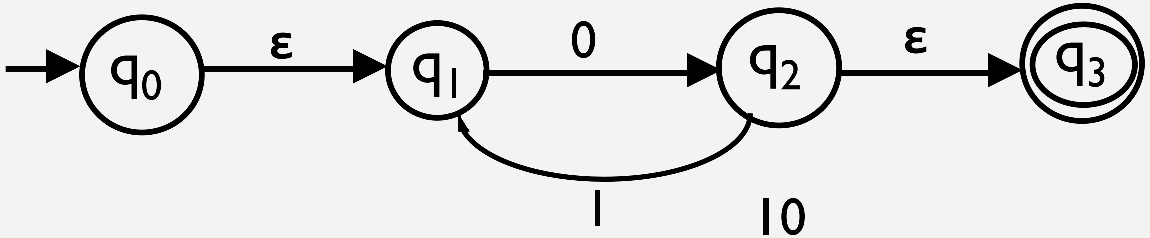
CONVERSION OF FINITE AUTOMATA TO RE:

2. Convert Following to R.E

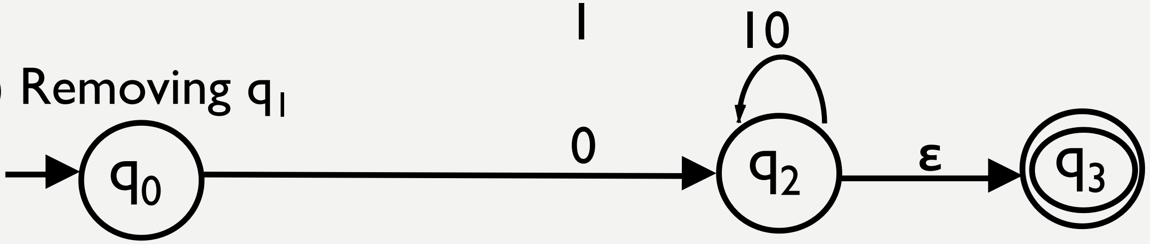


Solution:

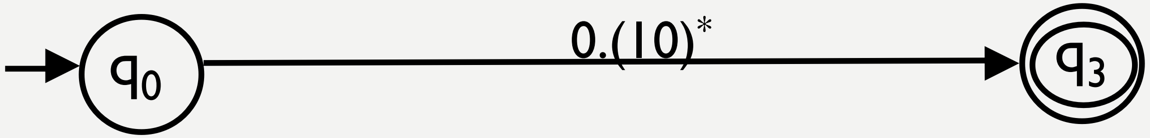
i)Since there is incoming edge in initial state, outgoing edge from final state



ii) Removing q_1



iii) Removing q_2



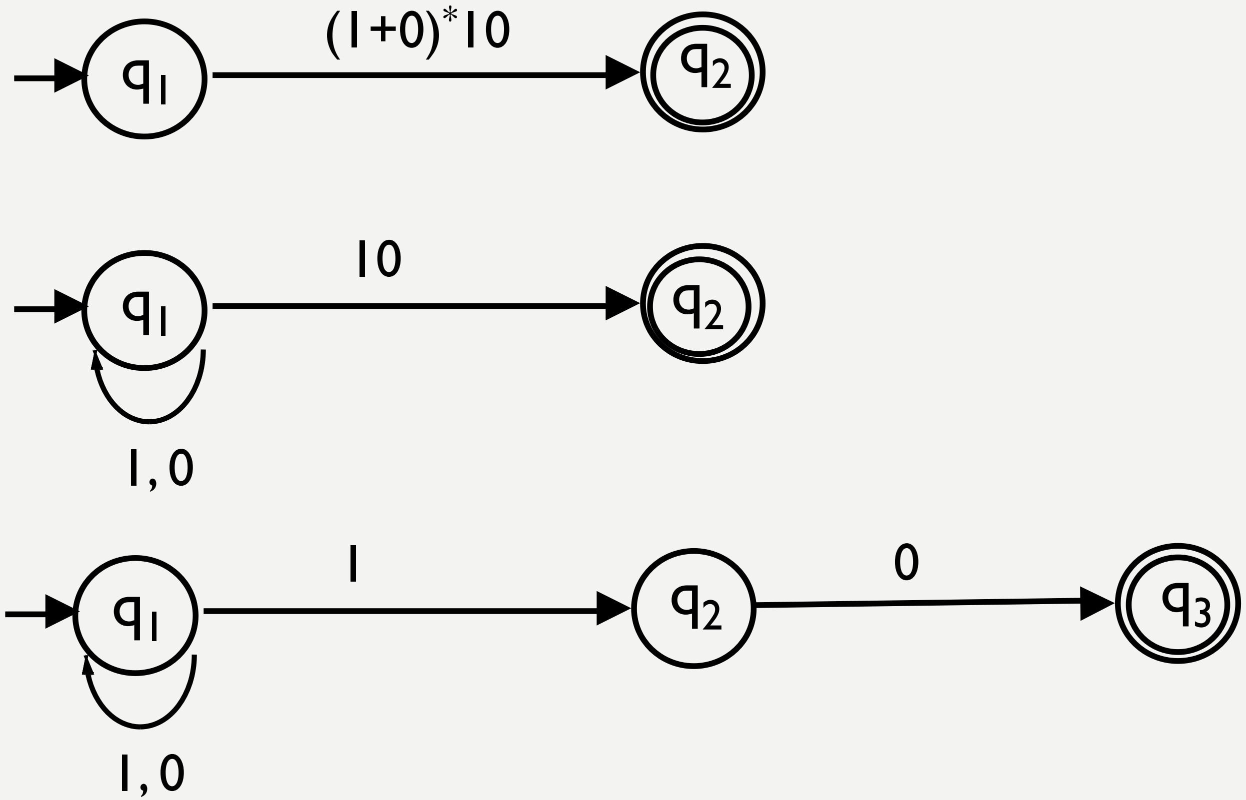
The Required R.E is:

$0.(10)^*$

CONVERSION OF RE TO FINITE AUTOMATA :

1. Convert Following to Finite Automata

Q.a $(1+0)^{*}10$



1

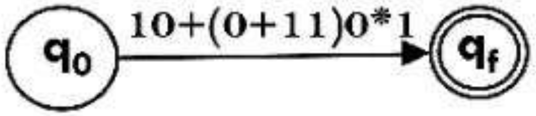
This is the required NFA

CONVERSION OF FINITE RE TO AUTOMATA :

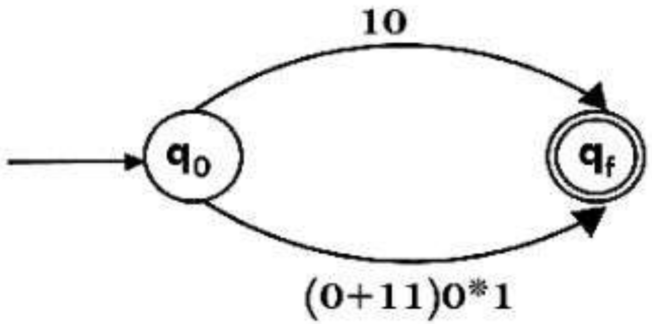
Design a FA from given regular expression 10 + (0 + 11)0* 1.

Solution: First we will construct the transition diagram for a given regular expression.

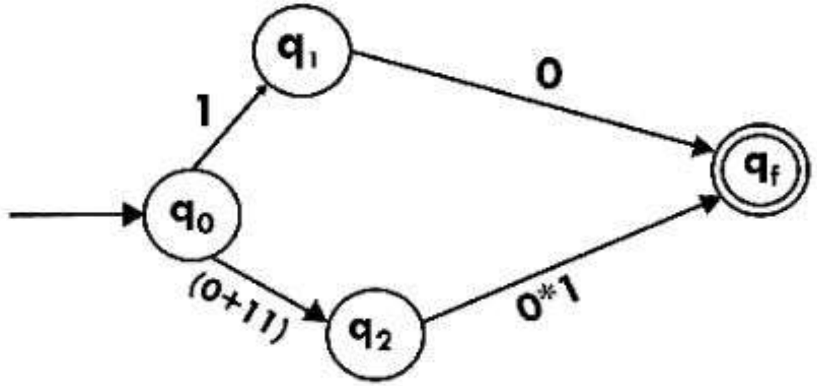
Step 1:



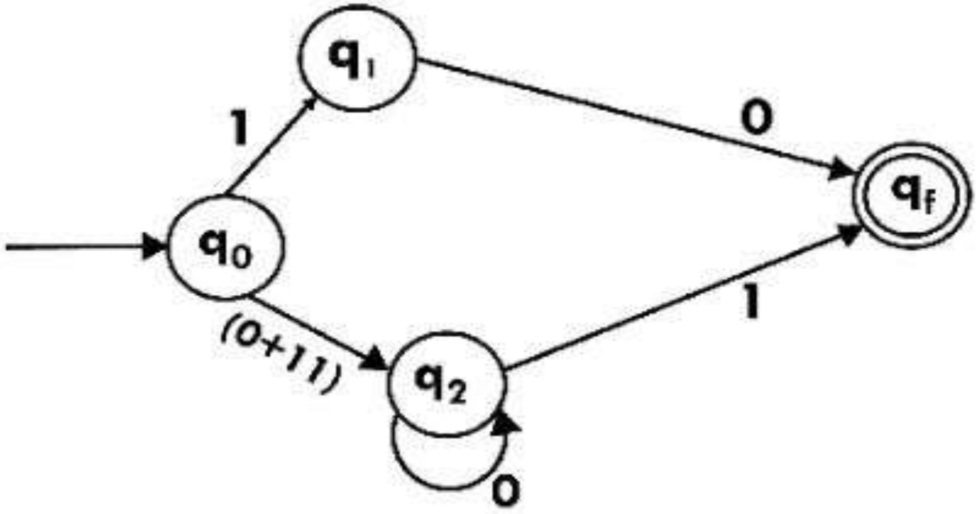
Step 2:



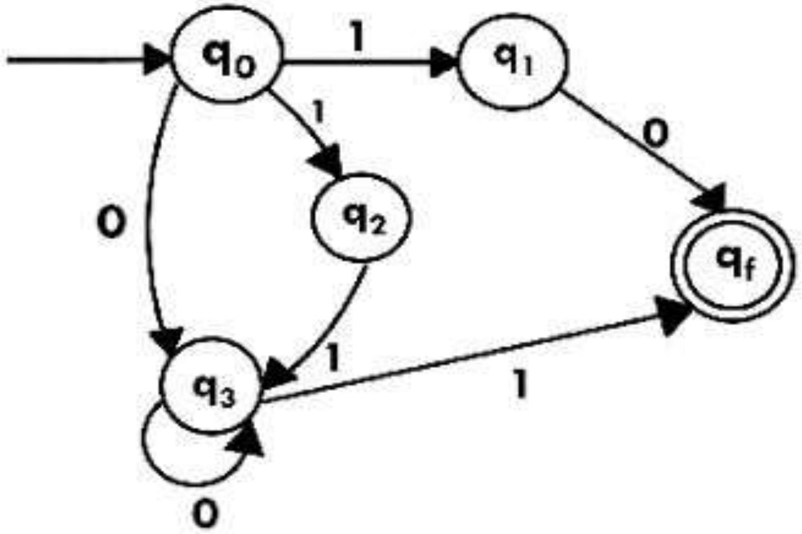
Step 3:



Step 4:



Step 5:



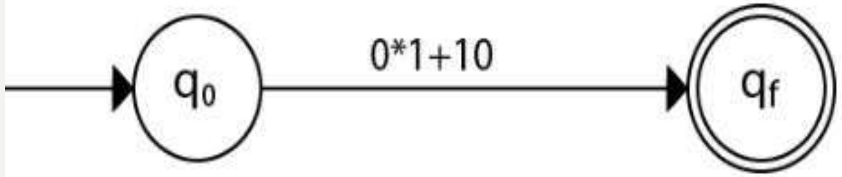
CONVERSION OF FINITE RE TO AUTOMATA :

Construct the FA for regular expression 0*1 + 10.

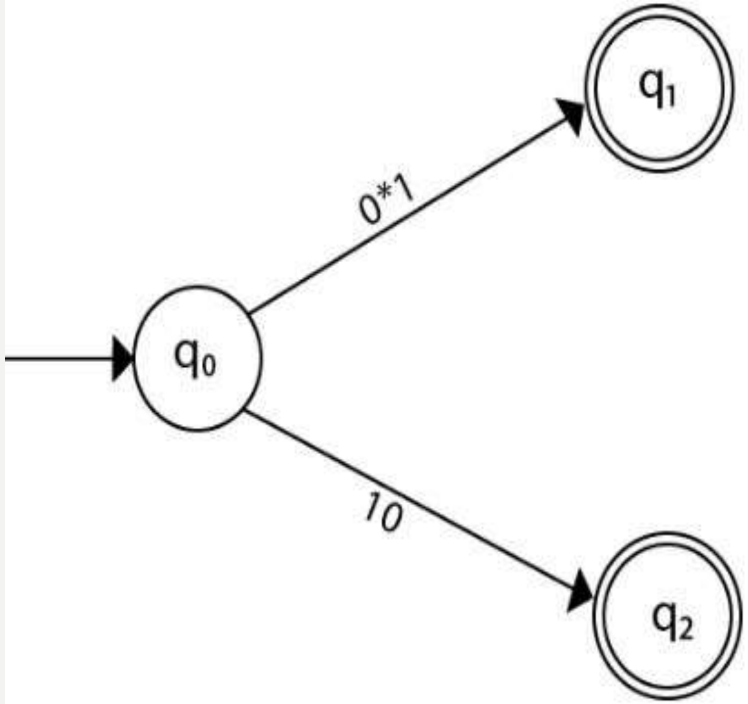
Solution:

We will first construct FA for R = 0*1 + 10 as follows:

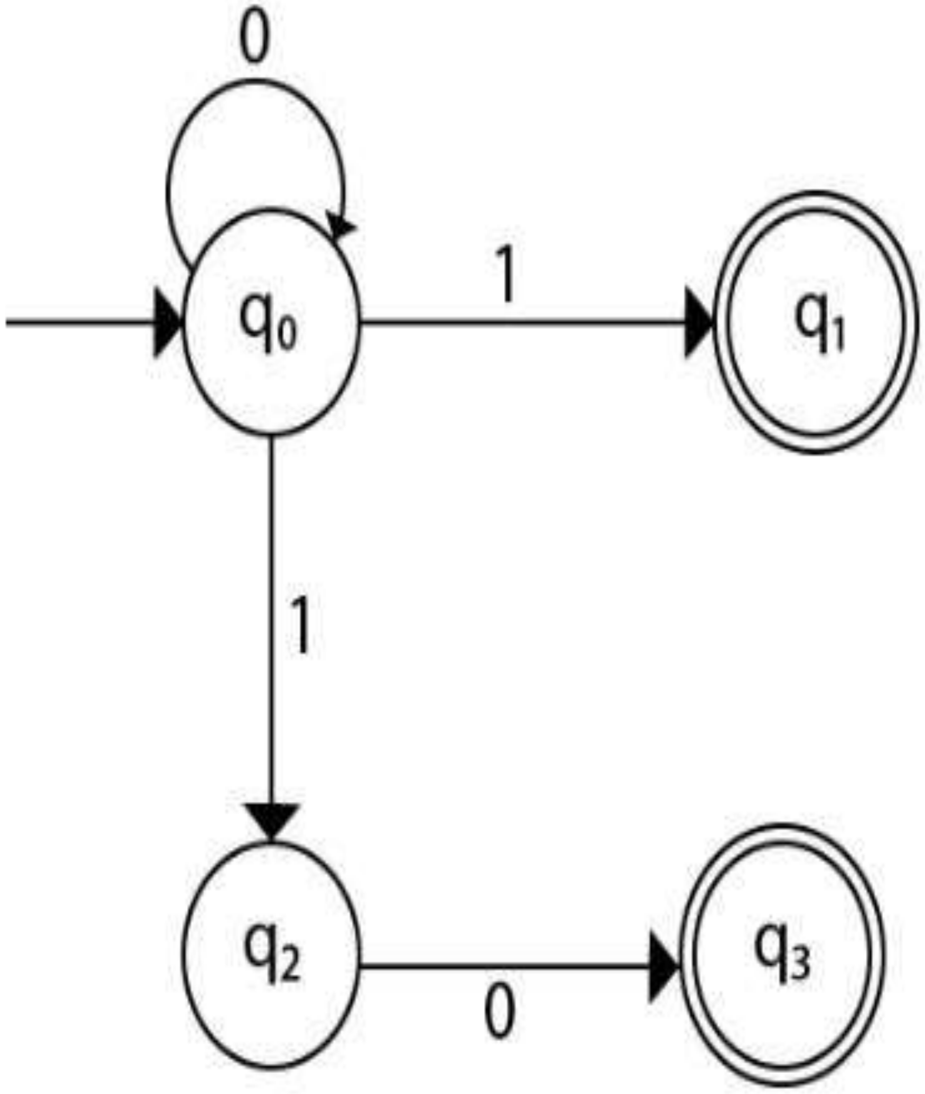
Step 1:



Step 2:



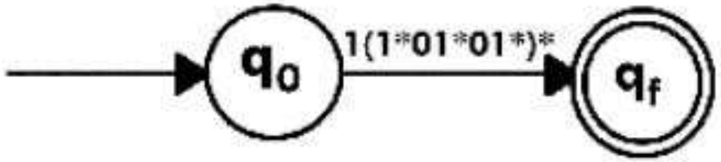
Step 4:



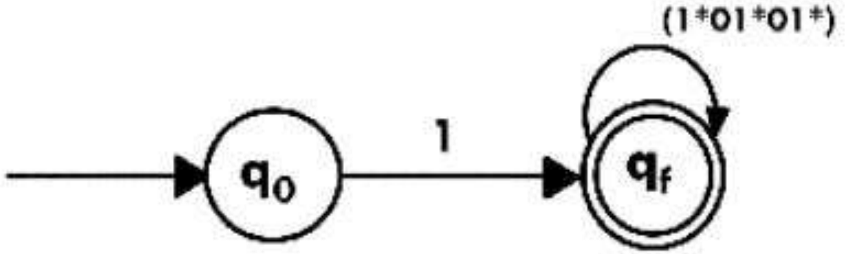
Design a NFA from given regular expression 1 (1* 01* 01*)*.

Solution: The NFA for the given regular expression is as follows:

Step 1:



Step 2:



Step 3: