> *A regular expression is a sequence of patterns that defines a string.*

The language accepted by finite automata can be easily described by simple expressions called regular expressions.

Operators of Regular expression

The definition of regular expression includes three basic operators:

1. Union
2. Concatenation
3. Closure

1. Union: If p and q are regular expressions, then p+q is a regular expression denoting the union of L(p) and L(q), that is, L(p+q) = L(p) U L(q)2.

2. Concatenation: If p and q are regular expressions, then p.q is a regular expression denoting the concatenation of L(p) and L(q), that is, L(pq) = L(p) L(q).

3. Closure: If p is regular expression, then so is p*, denoting the closure of L(p), that is L(p*) = (L(p))*.

Some regular expressions and its language

Regular expression	Language

r=a	L(r) = {a}
r = ab	L(r) = {ab}
r = a+b	L(r) = {a, b}
r = a*	L(r) = {∈, a, aa, aaa, ...}
r = ab*	L(r) = {a, ab, abb, abbb,....}
r = (ab)*	L(r) = {∈, ab, abab, ababab, ...}
r = a(a+b)	L(r) = {aa, ab}

a* VS a^+

a* = a power * means, a may not exist or may exist.

a^+ = a power + means, a exist atleast once.

Characterstics of regular expression

1. Regular expression is language defining notation in terms of algebraic description.
2. The languages accepted by finite automata, or regular language, is easily described by simple expressions called regular expressions.
3. It is more precise and formal as compared to any natural language.
4. In contrast to the transition graph, regular expressions can be conveniently written out in a line from left to right.
5. Main two areas of application of regular expression are:
    - Lexical analysis (compilers) and

- text editors.

Regular Expression examples:

Example 1: Let Σ = {a, b}. Write regular expression to define language consisting of strings w such that, w contains only a’s or only b’s of length zero or more.

Solution: r = a* + b*

Example 2: Let Σ = {a, b}. Write regular expression to define language consisting of strings w such that, w is of length one or more and contains only a’s or only b’s. r = a+ + b+

Solution: $r = a^+ + b^+$

Example 3: Let Σ = {a, b}. Write regular expression to define language consisting of strings w such that, w contains zero or more a’s followed by zero or more b’s

Solution: r = a*b*

Example 4: Let Σ = {a, b}. Write regular expression to define language consisting of strings w such that, w of length even

Solution: r = [(a + b) (a + b)]*

More Example

Related Posts:

1. Definition of Deterministic Finite Automata
2. Notations for DFA
3. How do a DFA Process Strings?
4. DFA solved examples
5. Definition Non Deterministic Finite Automata
6. Moore machine
7. Mealy Machine
8. Regular Expression Examples
9. Arden’s Law
10. NFA with ∈-Moves
11. NFA with ∈ to DFA Indirect Method
12. Define Mealy and Moore Machine
13. What is Trap state ?
14. Equivalent of DFA and NFA
15. Properties of transition functions
16. Mealy to Moore Machine

17. Moore to Mealy machine
18. Diiference between Mealy and Moore machine
19. Pushdown Automata
20. Remove ∈ transitions from NFA
21. TOC 1
22. Diiference between Mealy and Moore machine
23. RGPV TOC What do you understand by DFA how to represent it
24. What is Regular Expression
25. What is Regular Set in TOC
26. RGPV short note on automata
27. RGPV TOC properties of transition functions
28. RGPV TOC What is Trap state
29. DFA which accept 00 and 11 at the end of a string
30. CFL are not closed under intersection
31. NFA to DFA | RGPV TOC
32. Moore to Mealy | RGPV TOC PYQ
33. DFA accept even 0 and even 1 |RGPV TOC PYQ
34. Short note on automata | RGPV TOC PYQ
35. DFA ending with 00 start with 0 no epsilon | RGPV TOC PYQ
36. DFA ending with 101 | RGPV TOC PYQ
37. Construct DFA for a power n, n>=0 || RGPV TOC
38. Construct FA divisible by 3 | RGPV TOC PYQ
39. Construct DFA equivalent to NFA | RGPV TOC PYQ
40. RGPV Define Mealy and Moore Machine
41. RGPV TOC Short note on equivalent of DFA and NFA
42. RGPV notes Write short note on NDFA
43. Minimization of DFA

44. Construct NFA without ∈
45. CNF from S–>aAD;A->aB/bAB;B->b,D->d.
46. NDFA accepting two consecutive a’s or two consecutive b’s.
47. Regular expresion to CFG
48. Regular expression to Regular grammar
49. Grammar is ambiguous. S → aSbS|bSaS|∈
50. leftmost and rightmost derivations
51. Construct Moore machine for Mealy machine
52. RGPV TOC PYQs
53. Introduction to Automata Theory
54. Design a NFA that accepts the language over the alphabet, Σ= {0, 1,2} where the decimal equivalent of the language is divisible by 3.