Regular Expression

Theory Of Automata

Defining Languages (Regular Expression)

- Regular Expression (RE) is one of the language defining method.
- Regular Expression (RE) represented in terms of strings.
- In RE, a^* means zero or more occurrence of a, where as a^+ means one or more occurrence of a.
- Same concept for a^* and a^+ as of Kleene star and Kleene plus.

Regular Expression

As discussed earlier

- a^* generates Λ, a, aa, aaa, …
- a^+ generates a, aa, aaa, aaaa, …,

- Example: Consider alphabet Σ={a}, language L, Made from given alphabet is L = {Λ, a, aa, aaa, aaaa, …}. What will be the RE of this language.

- Similarly, for L_2 = {a, aa, aaa, aaaa, …} over alphabet Σ={a}. What will be RE.

Regular Expression

- Example: Write a RE for string that start from “a” and contain any “b” letter defined over alphabet $\Sigma=\{a,b\}$.
- Hint (According to given condition language must start from letter “a” and contain any no of ”b” letter)
- So, L could be {a, ab, abb, abbb, abbbb, ……}
- RE = ab^*

- Example: Write a RE for string that start from “a” and contain at least one “b” letter defined over alphabet $\Sigma=\{a,b\}$.
- So, L could be {ab, abb, abbb, ….}
- RE = ab^+

Regular Expression

- Now consider another language L, consisting of all possible strings, defined over $\Sigma = \{a, b\}$. This language can also be expressed by the regular expression

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

- Now consider another language L, of strings having exactly double a, defined over $\Sigma = \{a, b\}$, then it's regular expression may be

$$b^*aab^*$$

Regular Expression

- Write a RE for string that contains "a" or "b" defined over alphabet $\Sigma = \{a, b\}$.
- Hint(a or b) Also, when we have OR word in RE, we consider union operation and can be represented by "+" symbol.
- So, RE = ?

Class Task (Regular Expression)

- Write a RE for string that contains at least one “a” **OR** at least one “b” defined over $\Sigma = \{a, b\}$.

Class Task (Regular Expression)

- Write a RE for string that contains at least one "a" **OR** at least one "b" defined over $\Sigma = \{a, b\}$.
- Solution: RE = $(a+b)^+$

Regular Expression

- Consider language L, of even length, defined over $\Sigma = \{a, b\}$, then it's regular expression may be

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

- Now consider another language L, of odd length, defined over $\Sigma = \{a, b\}$, then it's regular expression may be

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

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

Remarks

- It may be noted that a language may be expressed by more than one regular expressions.
- While given a regular expression there exist a unique language generated by that regular expression.

Regular Expression

- Example:
  - Consider the language, defined over $\Sigma=\{a , b\}$ of words having at least one a, may be expressed by a regular expression $(a+b)^*a(a+b)^*$.

  - Consider the language, defined over $\Sigma = \{a,b\}$ of words having at least one a and one b, may be expressed by a regular expression $(a+b)^*a(a+b)^*b(a+b)^*+ (a+b)^*b(a+b)^*a(a+b)^*$.

Regular Expression

- Consider the language, defined over $\Sigma=\{a, b\}$, of words starting with double a and ending in double b then its regular expression may be $aa(a+b)^*bb$
- Consider the language, defined over $\Sigma=\{a, b\}$ of words starting with a and ending in b OR starting with b and ending in a, then its regular expression may be $a(a+b)^*b+b(a+b)^*a$

Regular Language

- **Definition:** The language generated by any regular expression is called a **regular language**.
- It is to be noted that if r_1, r_2 are regular expressions, corresponding to the languages L_1 and L_2 then the languages generated by r_1, r_2 are also regular languages.
- Example: If r_1=(aa+bb) and r_2=(a+b) then the language of strings generated by r_1+r_2, is also a regular language, expressed by (aa+bb)+(a+b)

Regular Language

All finite languages are regular

Example:

- Consider the language L, defined over Σ={a,b}, of strings of length 2, **starting with a**, then
- L={aa, ab}, may be expressed by the regular expression aa+ab. Hence L, by definition, is a regular language.

Finite Automata

- So far, we have studied Different ways of defining languages i.e., Descriptive, Recursive, Regular Expression.
- Now we move towards Finite Automata (FA).
- In Finite Automata we can represent language in form of diagram or graph.
- Finite Automata is also known as:
  - Finite Machine (FM)
  - Finite Automatic Machine (FAM)
  - Finite State Machine (FSM)

Finite Automata

Method 4 (Finite Automaton)

Definition:

A **Finite automaton (FA)**, is a collection of the followings

1. Finite number of states, having one initial and some (maybe none) final states.

2. Finite set of input letters (Σ) from which input strings are formed.

3. Finite set of transitions *i.e.,* for each state and for each input letter there is a transition showing how to move from one state to another.

Finite Automata

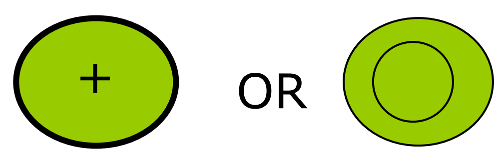
- Also, A finite automaton is a 5-tuple (Q, Σ, δ, q0, F), where:

1. Q is a finite set called the states (e.g., q0, q1, q2, q3,…..)
2. Σ is finite set called the alphabet (e.g., Σ = {a,b})
3. δ : Transition / Movement of data in form of arrows
4. q0: is the start state (initial state)
5. F (final states)

Finite Automata

1. q0 (start state or initial state) of Finite Automata (FA)
2. In every FA, there will be only one initial state
3. It is represented by represent input.

4. At final state, machine may stop or may not stop depend on your language (input)
5. It is represented by:

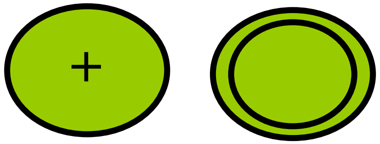


Finite Automata

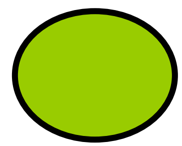
Final State

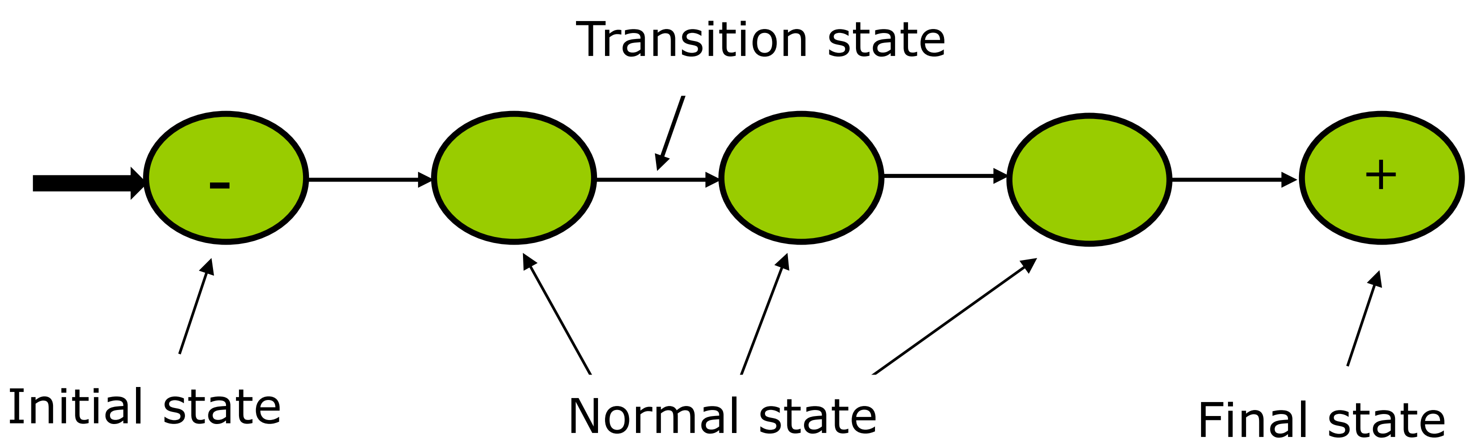
- In FA the minimum number of final state should be 1 and maximum number of final state can be more than 1.

- It means we may have more than q final state in FA.
- At final state, machine may stop or may not stop depending upon language (input).
- It is represented by



Finite Automata

1. Q (states/normal states) these are the states other then initial and final states.
2. These are states from which our FA neither start nor finish.
3. It is represented by: 
4. δ: transition / movement : transition or movement is represented with sign.



Types of Finite Automata (FA)

- Finite Automata without output
  - Deterministic Finite Automata (DFA).
  - Non-Deterministic Finite Automata (NFA or NDFA).
  - Non-Deterministic Finite Automata with epsilon moves (e-NFA or e-NDFA).
- Finite Automata with Output
  - Moore machine.
  - Mealy machine.

Will be discussing in future lectures

Finite Automata (FA)

Now Back to Automata

- Also, A finite automaton is a 5-tuple (Q, Σ, δ, q0, F), where:

1. Q is a finite set called the states (e.g., q0, q1, q2, q3,…..)
2. Σ is finite set called the alphabet (e.g., Σ = {a,b})
3. δ : Transition / Movement of data in form of arrows
4. q0: is the start state (initial state)
5. F (final states)

Example

- $\Sigma = \{a,b\}$

  States: x, y, where x is both initial and final state.

  Transitions:

  1. At state x reading a or b go to state y.
  2. At state y reading a or b go to state x.

Example Continued …

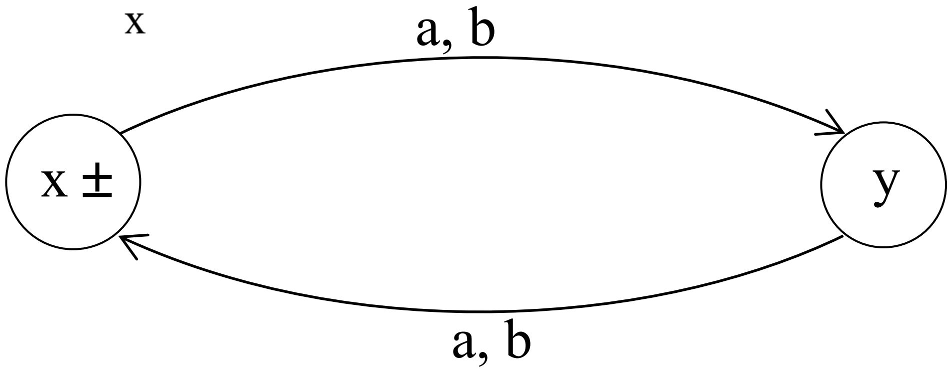
- These transitions can be expressed by the following transition table

Old States	New States	
	Reading a	Reading b
x ±	y	y
y	x	x

Example Continued …

- It may be noted that the previous transition table may be depicted by the following transition diagram.

Old States	New States	
	Reading a	Reading b
x ±	y	y
y	x	x



Example Continued …

Thank you, Questions?