

# (Theory of Computation)



## Topics

## GATE Weightage

Finite Automata

7-9 Marks

Regular Expression & Regular Language

Grammar

Pushdown Automata

Turing Machine

Undecidability

## ( ABOUT ME )



More than 12 years GATE teaching experience

Mentored more than 20,000 student

Work as a freelancer many reputed institute in across all India

Qualified GATE(99.24)&UGC NET  
In Computer Science



**SHAILENDRA SINGH**

# (Theory of Computation)



Study about different types of Computation models.

OR

“Study about Mathematical Model of Computer”

**Basics:**  
**Symbol:**



**Alphabet ( $\Sigma$ ):**



# (Theory of Computation)

**String:**



**Null String ( Empty String ) :**



**Empty Language:**

**Reverse of a String:**





## (Theory of Computation)



**Substrings:** Taking zero or more consecutive symbols of a string



**Prefix of a String:** Taking zero or more symbols of a string from left to right

**Suffix of a String:** Taking zero or more symbols of a string from right to left



**Palindrom:**

**Cocatenation:**



**Power of  $\Sigma$ :**



**Kleen Star Closure:**



**Positive Closure:**



# (Theory of Computation)

**Grammar:**





# (Theory of Computation)

**Language:**





Ex-Consider the languages  $L_1 = \emptyset$  and  $L_2 = \{a\}$ . Which one of the following represents  $L_1 L_2 U L_1^*$ ?

- A)  $\{\epsilon\}$       B)  $a^*$       C)  $\{\epsilon, a\}$       D) None

**Complement:**



## Finite Automata:



**Block Diagram of FA:**





## **Application of Finite Automata:**

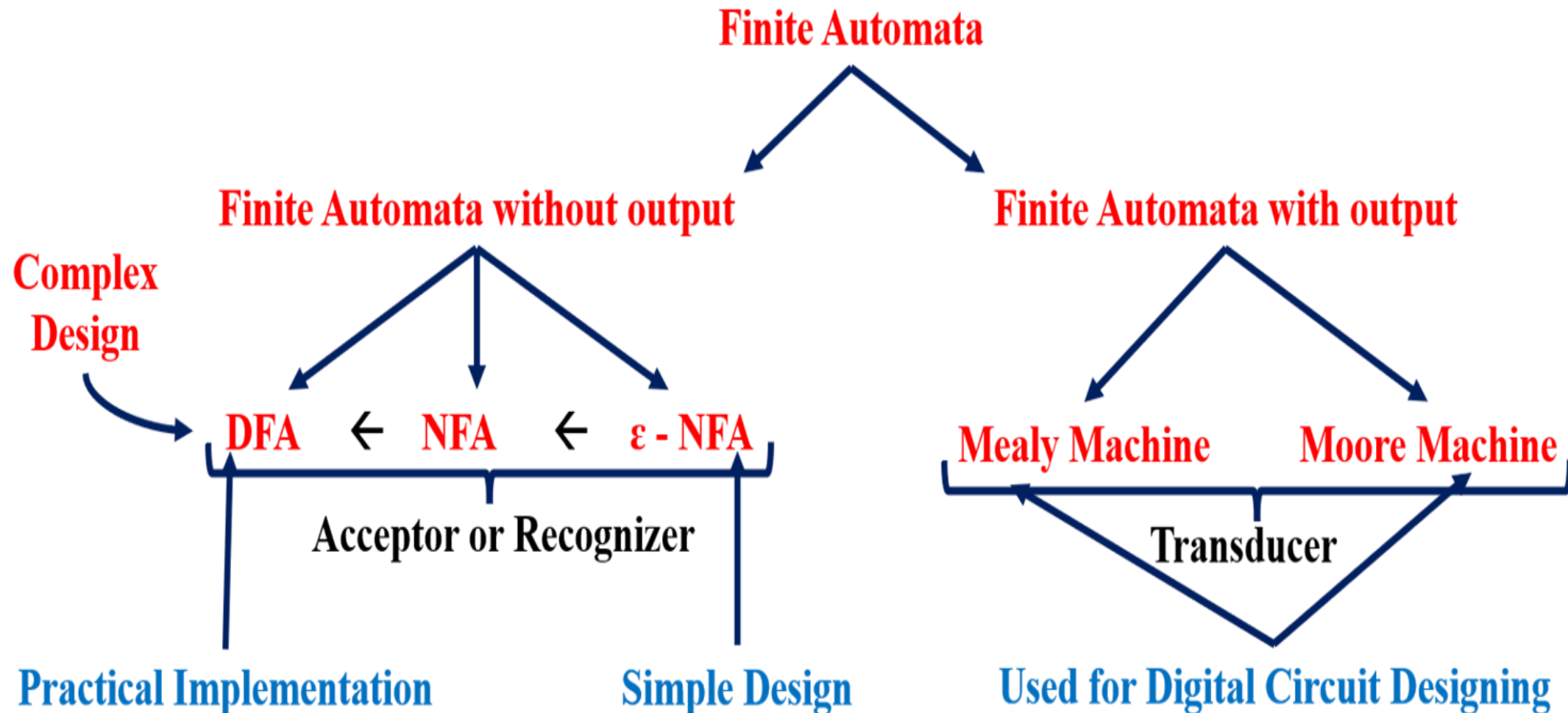
- 1- It is used in lexical Analysis of a compiler
- 2- It is used in the Design of Digital circuit( Combination and Sequential circuit) using Mealy and Moore machine
- 3- Used in Text Editors
- 4- Used in the implementation of spell checkers

## **Limitation of Finite Automata:**

- 1- It can not count and store



## Types of Finite Automata:



## **Deterministic Finite Automata (DFA):**







## Remarks:

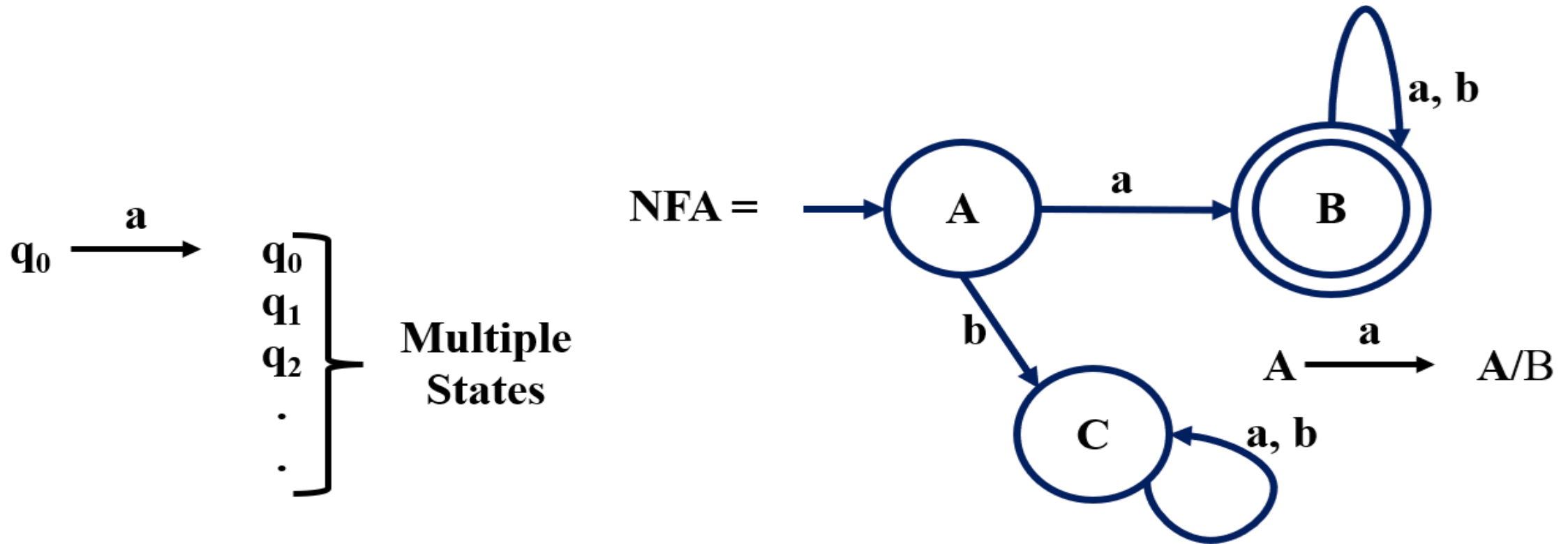
- It has only one unique next state.
- It has no choice or randomness.
- Given the current state we know what the next state will be.
- Dead Configuration not allowed because DFA is complete system.





## Non-Deterministic Finite Automata (NFA):

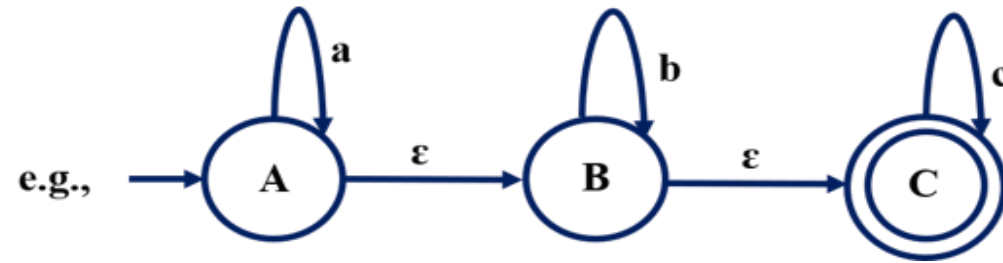
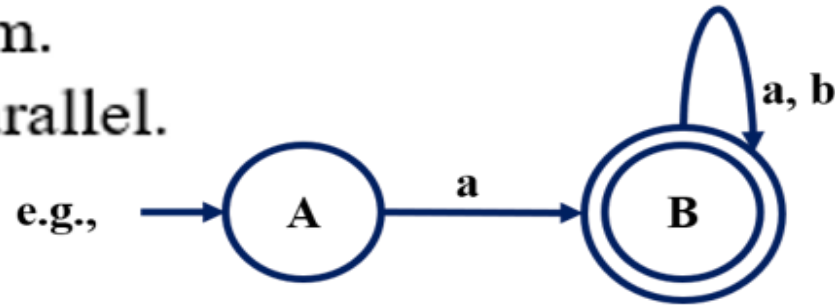
In NFA, given the current state there could be multiple next states



## (Theory of Computation)



- The next state may be chosen at random.
- All the next states may be chosen in parallel.
- Dead configuration allowed in NFA
- $\epsilon$  - transition allowed in NFA



## (Theory of Computation)

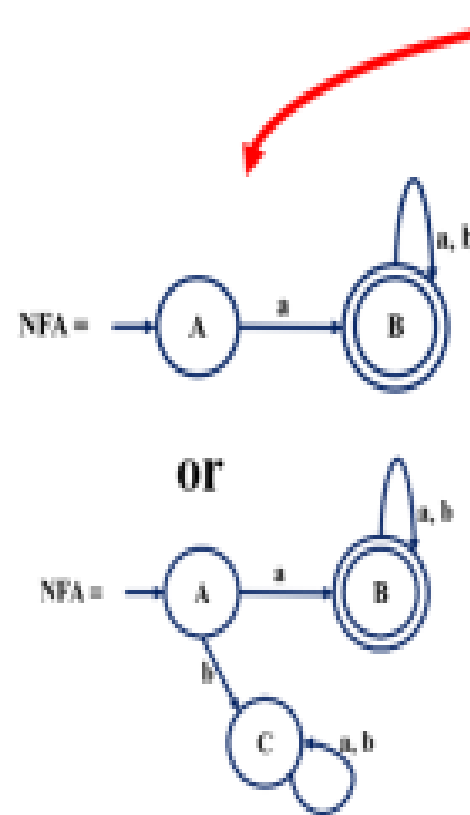
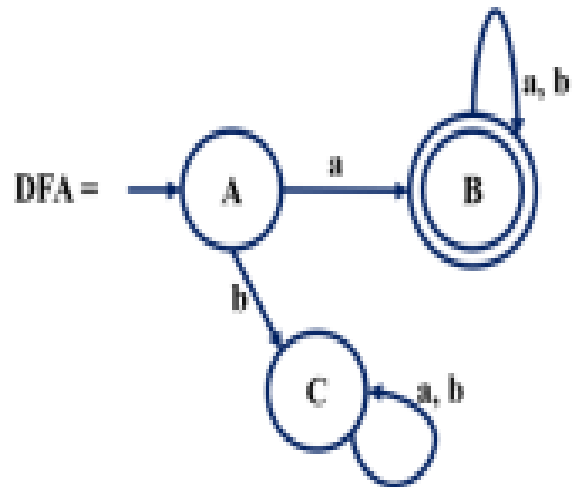
**Note: Why Dead configuration allowed in NFA, b/c NFA transition function  $\delta: Q \times \Sigma \rightarrow 2^Q$**



# (Theory of Computation)

Example:

Set of all strings start with a



**IMPORTANT**

**Note: Minimum number of states in finite automata always consider NFA**



## (Theory of Computation)

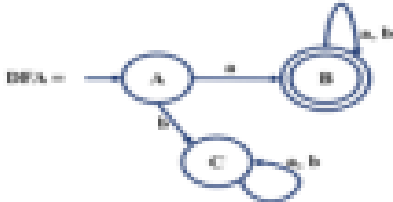



- Note:**
1. NFA is more general machine as compared to DFA.
  2. Every DFA is a NFA but every NFA is not a DFA, but there is an equivalent DFA for every NFA.  
i.e  $P(\text{NFA}) = P(\text{DFA})$
  3. Number of states in NFA =  $m$ , then      Number of states in DFA =  $n$ ,  $1 \leq n \leq 2^m$

# (Theory of Computation)



## Difference between DFA and NFA:

DFA	NFA
1. No Choice	1. Choice allowed
2. $\epsilon$ - moves not allowed	2. $\epsilon$ - moves allowed
3. Dead configuration not allowed	3. Dead configuration is allowed
4. Dead state maybe required	4. Dead state is not required
5. Number of next state is one	5. Number of next state can be zero, one, or more
	
6. NFA is easy to construct	6. DFA is relatively difficult to construct
7. Conversion of NFA to DFA has at most $2^Q$ states i.e., $\leq 2^Q$	7. If Q – state in NFA
8. Implementation of DFA require more space	8. Implementation of NFA requires less space then DFA
9. The total time required for running any input string in DFA is less	9. The total time required for running any input string in NFA is larger.
10. $\delta: Q \times \Sigma \rightarrow Q$	10. $\delta: Q \times \Sigma \rightarrow 2^Q$ or $P(Q)$
11. All DFA are NFA	11. Not all NFA are DFA



## Representation of Finite Automata:



**Acceptance by Finite Automata:**





## (Theory of Computation)

**DFA Designing: if  $\Sigma = \{a, b\}$**

**Q1. Design DFA for empty language**

**Q2. Design DFA for empty string**



**Q3. Set of all strings start with abb, over  $\Sigma$ . Design DFA.**



**Q4. Set of all strings end with abb, over  $\Sigma$ , Design DFA**



## (Theory of Computation)

**Q5. Set of all strings contain abb as a substring, over  $\Sigma$ , Design DFA**



**Q6. Set of all strings, Does not contain abb as a substring, over  $\Sigma$ , Design DFA**





## (Theory of Computation)

**Q7. Set of all strings, start and end with different symbol, over  $\Sigma$ , Design DFA.**



**Q8. Set of all strings, start and end with same symbol, over  $\Sigma$ , Design DFA.**



## (Theory of Computation)

**Q9. Design DFA, such that every a followed by at least two b over  $\Sigma$**





**Q10. Design DFA, over  $\Sigma$  for**

- (i) Even Length String
- (ii) Odd Length String

**Q11. Set of all strings, starts with either aa or bb, over  $\Sigma$**



**Q12. Set of all strings, ends with either aa or bb, over  $\Sigma$**





**Q13. Design a DFA for the set of strings over {a, b} such that length of the string W is**

- i)  $|W| = 2$  (Exact)
- ii)  $|W| \geq 2$  (at least)
- iii)  $|W| \leq 2$  (at most)



**Q14. Design DFA of the language containing the set of all strings over {a,b} in which 2<sup>nd</sup> symbol from LHS is 'a'**





**Q15. Design a DFA of the language containing the set of all strings over {a, b} in which 2<sup>nd</sup> symbol from RHS is 'a'**



**Q16. Design DFA for the set of string over  $\{a, b\}$  such that length of the string  $|W|$  is divisible by 3 i.e.,  $|W| \bmod 3 = 0$**

**OR**

**$L = \{W \mid |W| \bmod 3 = 0, w \in (a, b)^*\}$**

## (Theory of Computation)

**Q17. Design a DFA for the language,  $L = \{W \mid n_a(w) \bmod 2 = 0 \text{ and } n_b(w) \bmod 3 = 0, w \in (a, b)^*\}$ , means  $n_a(w)$  divisible by 2 and  $n_b(w)$  in  $w$  is divisible by 3**







**Q18. Design DFA over input alphabet  $\Sigma = \{0, 1\}$ , that accepts**

- i) Even number of 0's and Even number of 1's
- ii) Even number of 0's and Odd number of 1's
- iii) Odd number of 0's and Even number of 1's
- iv) Odd number of 0's and odd number of 1's

## (Theory of Computation)



**Q19. Design DFA for module – 3 machine, base binary i.e.,  $(\text{Mod} - 3)_2$**

**OR**

**Design DFA for  $L = \{s \mid d(s) \bmod 3 = 0, \text{ where } d(s) \text{ is decimal value of } s, s \in (0, 1)^*\}$**



**Q20. Design DFA for  $L = \{s \mid d(s) \bmod 5 = 0, \text{ where } d(s) \text{ is decimal value of } s \text{ and } s \in (0, 1)^*\}$**

**OR**

**$(\text{Mod } 5)_2$ : Modulo – 5 machine, Base is binary**

## (Theory of Computation)

**NOTE:** The minimum number of states in the DFA required to check the divisibility of a binary string is?







**Q21. Construct a minimal DFA that accepts all binary no. that are**

- (i) Divisible by 2 and 3**
- (ii) Divisible by 2 or 3**
- (iii) Divisible by 2 but not 3**
- (iv) Divisible by 3 but not 2**
- (v) Neither Divisible by 3 nor 2**



**Q22. Construct a minimal DFA over  $\{a,b\}$  that accepts all strings having**

- (i) At Least 2 a's and At Least 3 b's**
- (ii) At Least 2 a's and At Most 3 b's**
- (iii) At Most 2 a's and At Most 3 b's**
- (iv) Exact 2 a's and Exact 3 b's**