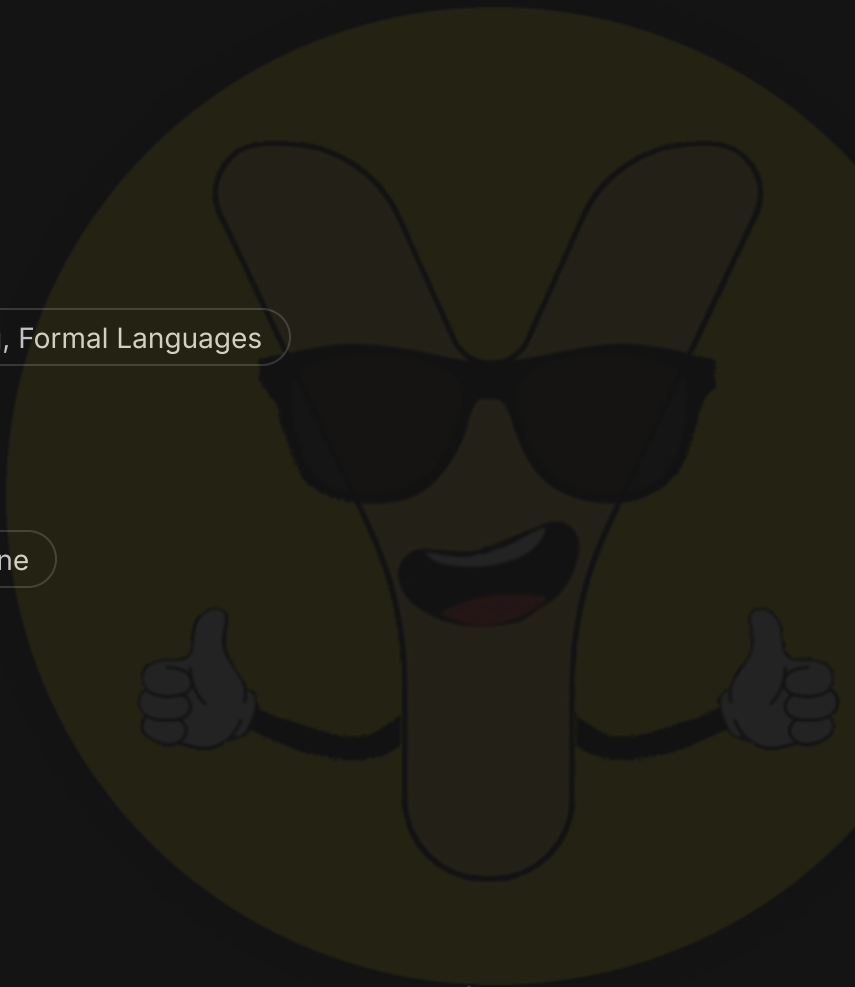
AKTU · 2ND YEAR · SEMESTER 4 · BCS402

Theory of Automata and Formal Languages

Unit 1 — Basic Concepts and Automata Theory

- Introduction to Theory of Computation
- Automata, Computability and Complexity
- Alphabet, Symbol, String, Formal Languages
- Deterministic Finite Automaton (DFA)- Definition, Representation, Acceptability of a String and Language
- Non Deterministic Finite Automaton (NFA)
- Equivalence of DFA and NFA
- NFA with ε-Transition
- Equivalence of NFA's with and without ε-Transition
- Finite Automata with output- Moore Machine, Mealy Machine



DECODED BY

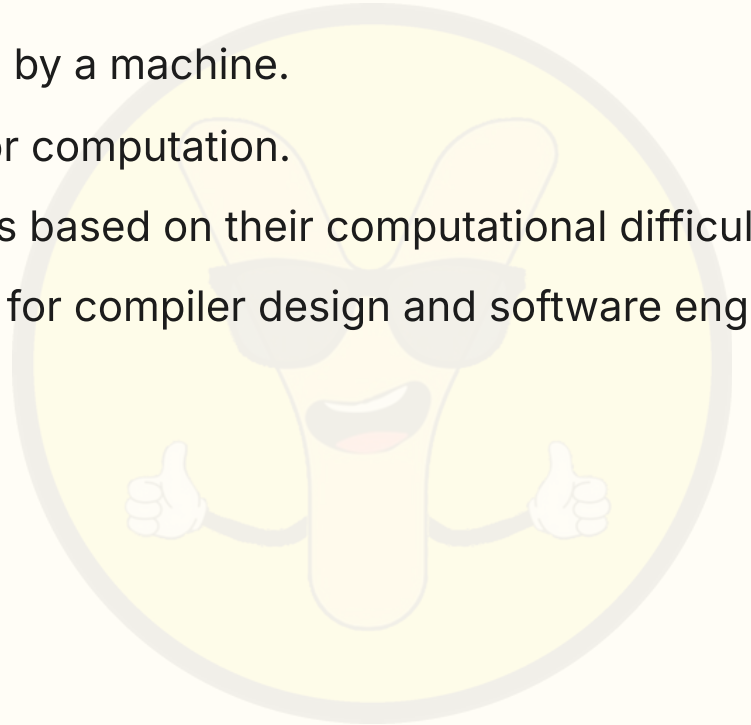
Chirag Jain

Founder, Ycotes · Gold Medalist in Artificial Intelligence, IIT Madras · taught 10,000+ students

Introduction to Theory of Computation

Theory of Computation (TOC) explores the fundamental capabilities and limitations of computers.

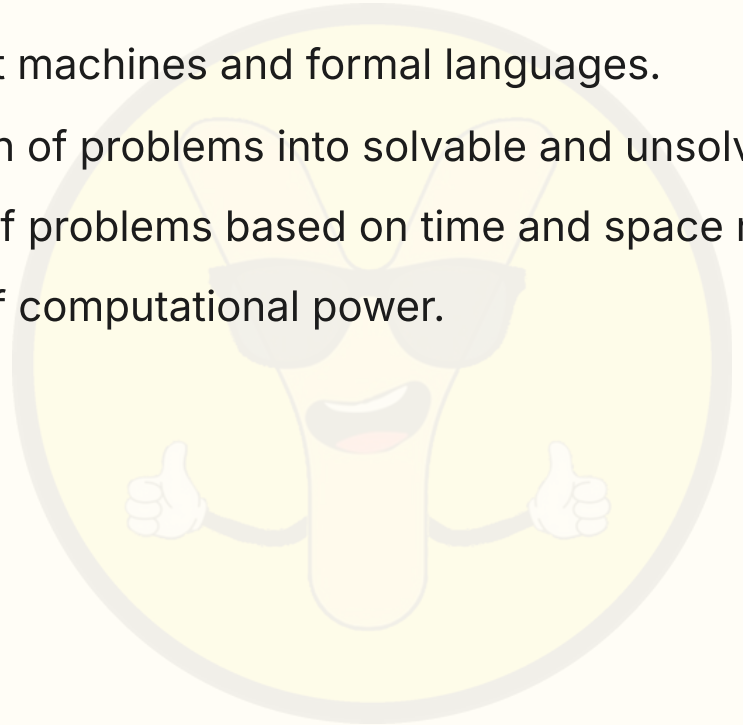
- It deals with what can be computed by a machine.
- It provides a mathematical model for computation.
- Core objective: To classify problems based on their computational difficulty.
- It acts as the theoretical foundation for compiler design and software engineering.



Automata, Computability, and Complexity

TOC is divided into three major branches.

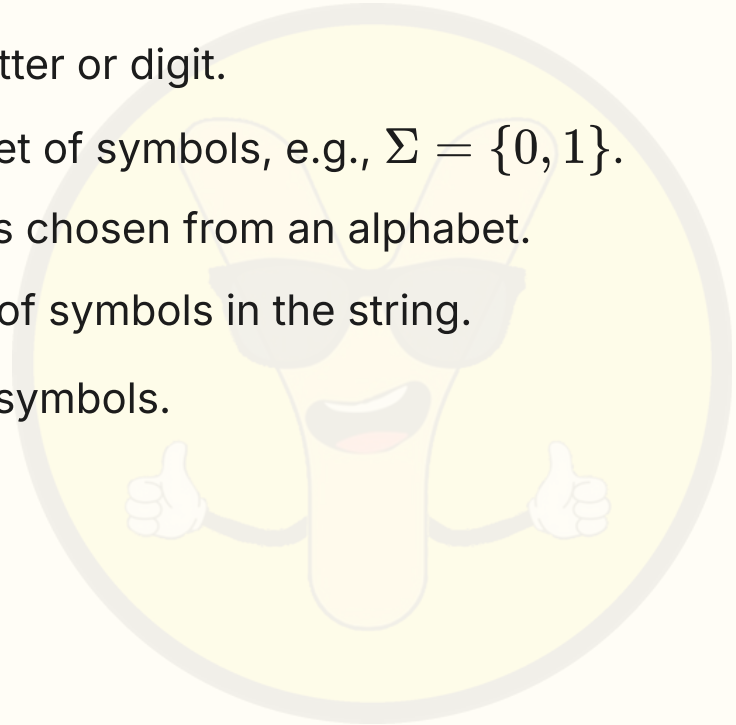
- Automata Theory: Study of abstract machines and formal languages.
- Computability Theory: Classification of problems into solvable and unsolvable categories.
- Complexity Theory: Classification of problems based on time and space requirements.
- These branches form a hierarchy of computational power.



Alphabet, Symbol, and String

Basic building blocks of formal languages.

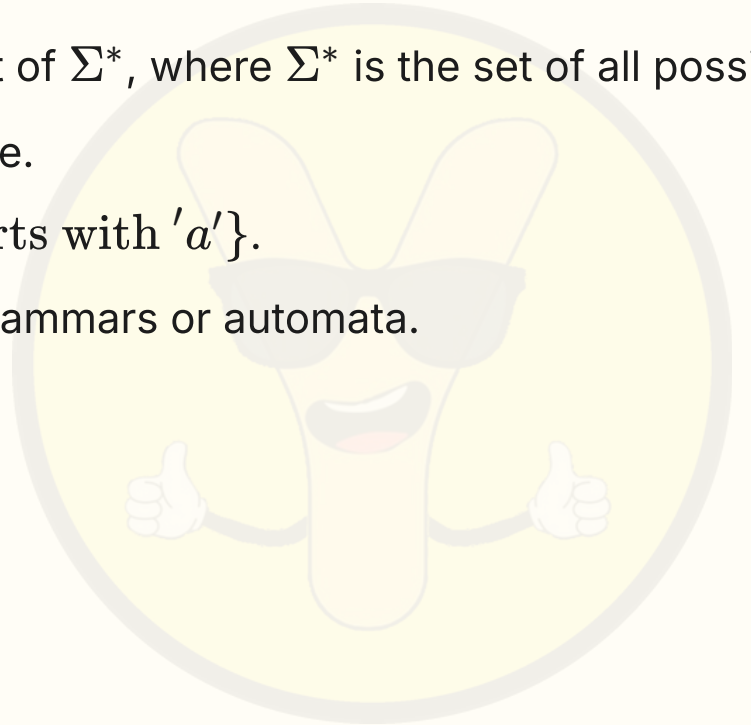
- Symbol: The smallest unit, e.g., a letter or digit.
- Alphabet (Σ): A finite, non-empty set of symbols, e.g., $\Sigma = \{0, 1\}$.
- String: A finite sequence of symbols chosen from an alphabet.
- Length of string ($|w|$): The number of symbols in the string.
- Empty string (ϵ): A string with zero symbols.



Formal Languages

A language is a set of strings over a given alphabet.

- Definition: A language L is a subset of Σ^*, where Σ^* is the set of all possible strings.
- The language can be finite or infinite.
- Example: $L = \{w \in \{a, b\}^* | w \text{ starts with } 'a'\}$.
- Formal languages are defined by grammars or automata.



Deterministic Finite Automaton (DFA)

DFA

A DFA is a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$.

- Q: Finite set of states.
- Σ: Finite set of input symbols.
- δ: Transition function $Q \times \Sigma \to Q$.
- q_0: Initial state.
- F: Set of final or accepting states.



DFA Representation and Acceptability

How a DFA processes input.

- Representation: Transition diagrams (graphical) or Transition tables (tabular).
- Acceptability: A string w is accepted if the path starting from q_0 ends in a state $f \in F$.
- DFA is deterministic: For every state and input symbol, there is exactly one transition.
- No ambiguity exists in the movement between states.

Non-Deterministic Finite Automaton (NFA)

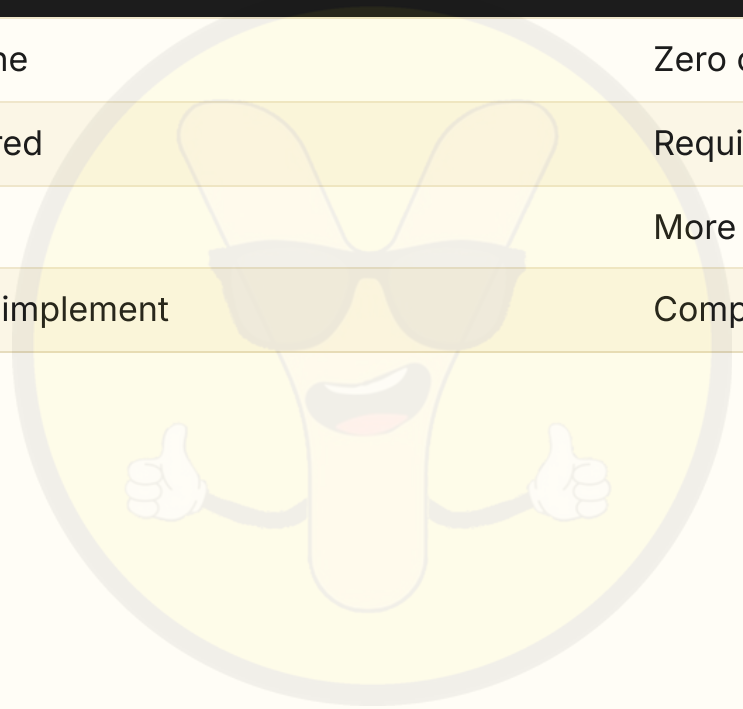
NFA

An NFA is similar to a DFA but allows multiple transitions for the same input.

- Transition function: $\delta : Q \times \Sigma \to 2^Q$.
- An NFA can move to zero, one, or more states for a single input.
- Acceptability: A string is accepted if at least one path leads to a final state.
- NFAs are often easier to design but equivalent in power to DFAs.

DFA vs NFA

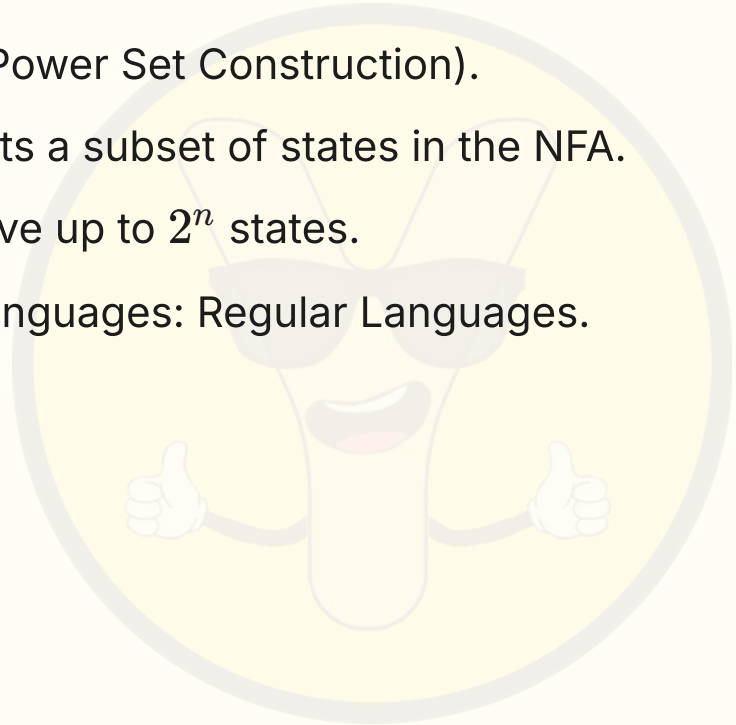
Feature	DFA	NFA
Transitions	Exactly one	Zero or more
Backtracking	Not required	Required
Memory	Less	More
Complexity	Simple to implement	Complex to implement



Equivalence of DFA and NFA

Every NFA can be converted into an equivalent DFA.

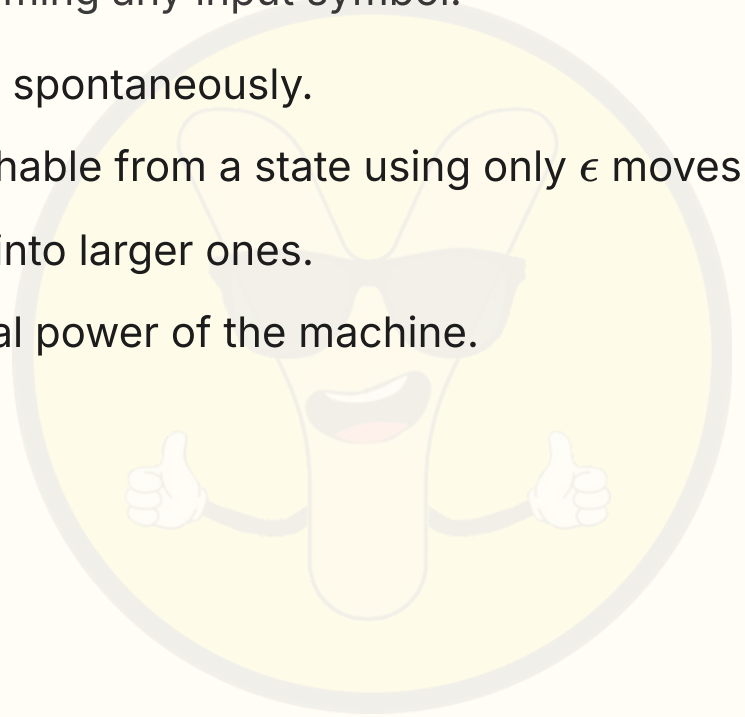
- Subset Construction Algorithm (or Power Set Construction).
- Each state in the new DFA represents a subset of states in the NFA.
- If NFA has n states, the DFA can have up to 2^n states.
- Both recognize the same class of languages: Regular Languages.



NFA with ϵ-Transition

NFA allowing transitions without consuming any input symbol.

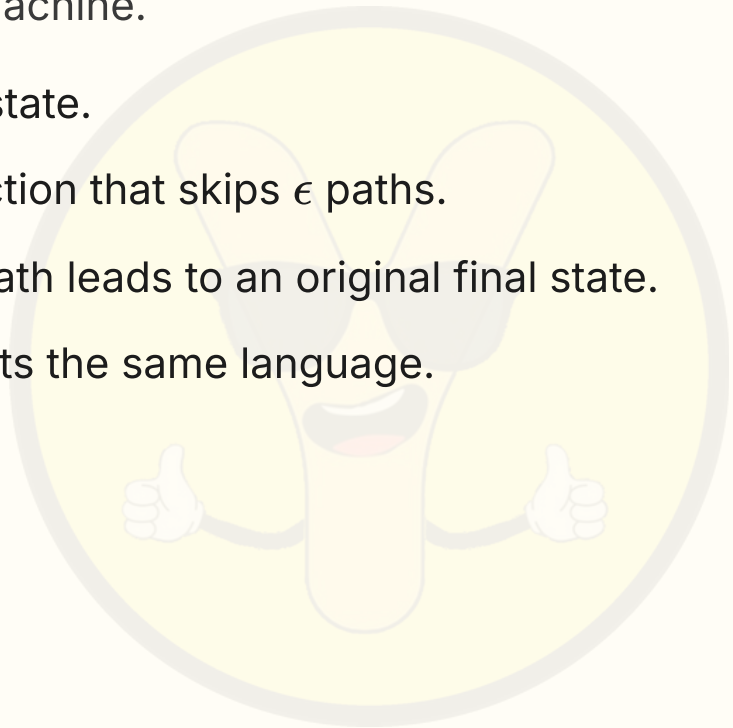
- Allows the machine to change state spontaneously.
- ϵ-closure: The set of all states reachable from a state using only ϵ moves.
- Useful for combining smaller NFAs into larger ones.
- Does not increase the computational power of the machine.



Equivalence: NFA with and without ϵ

Removing ϵ transitions simplifies the machine.

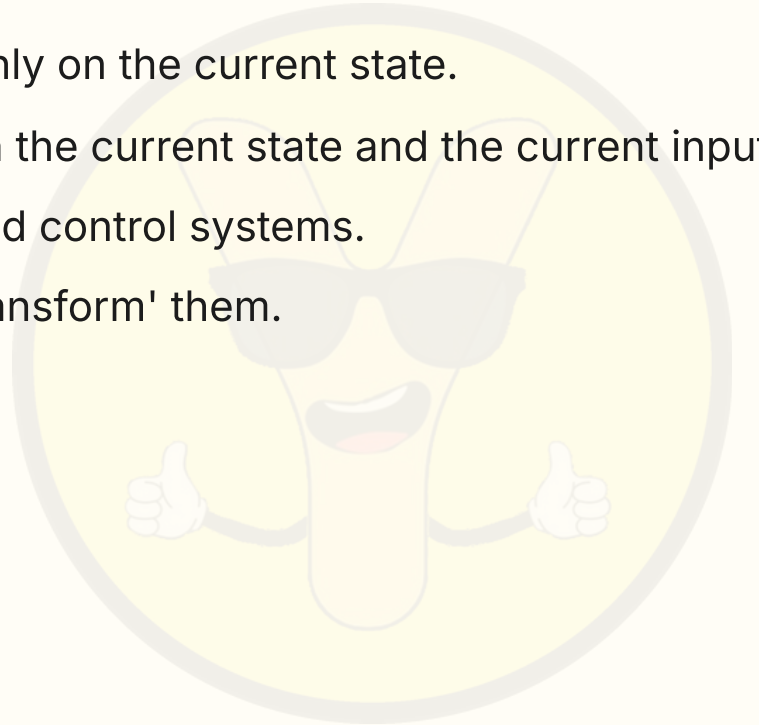
- Step 1: Find the ϵ-closure for each state.
- Step 2: Create a new transition function that skips ϵ paths.
- Step 3: Update final states if an ϵ-path leads to an original final state.
- Result: An NFA without ϵ that accepts the same language.



Finite Automata with Output

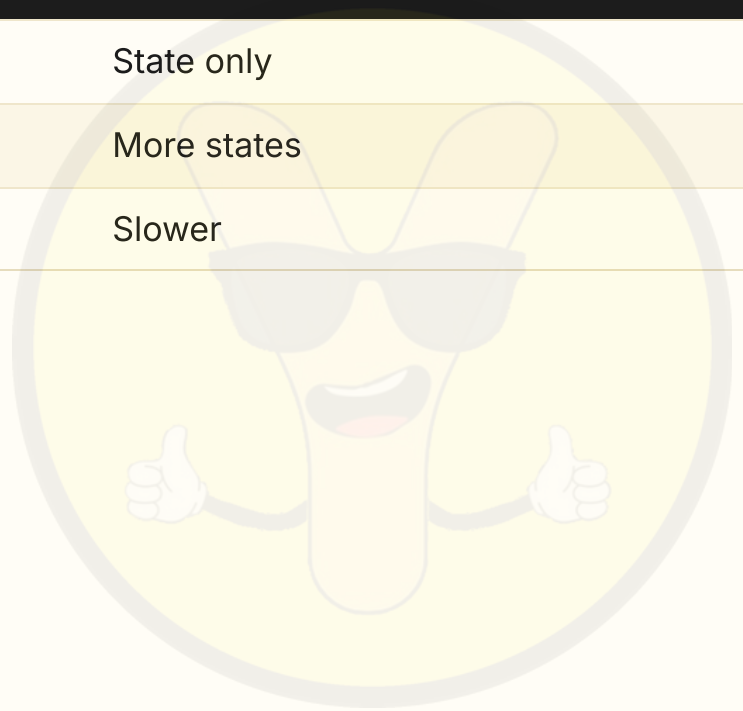
Machines that produce an output for every input sequence.

- Moore Machine: Output depends only on the current state.
- Mealy Machine: Output depends on the current state and the current input.
- Used in sequential circuit design and control systems.
- They do not 'accept' strings but 'transform' them.



Moore vs Mealy Machine

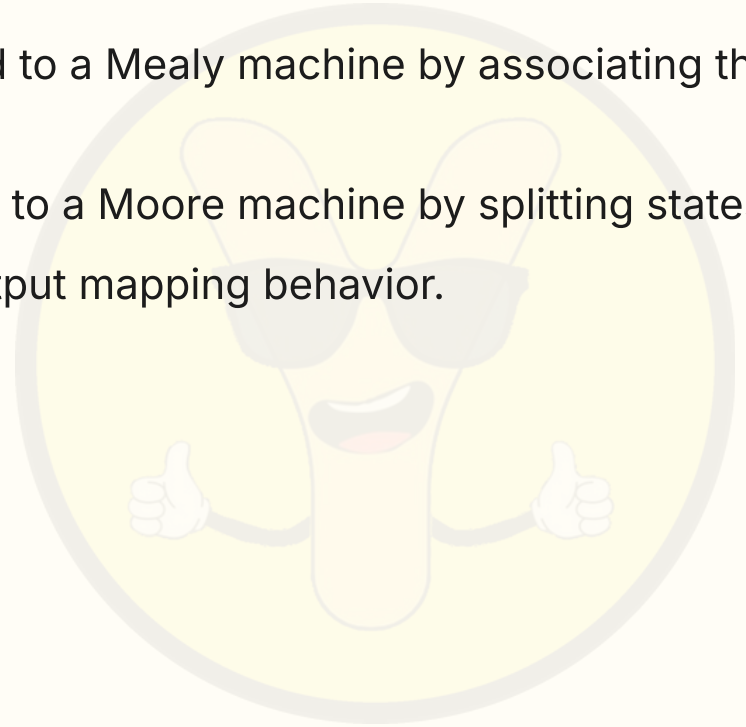
Feature	Moore Machine	Mealy Machine
Output depends on	State only	State and Input
Number of states	More states	Fewer states
Reaction	Slower	Faster



Equivalence of Moore and Mealy

Both machines are computationally equivalent.

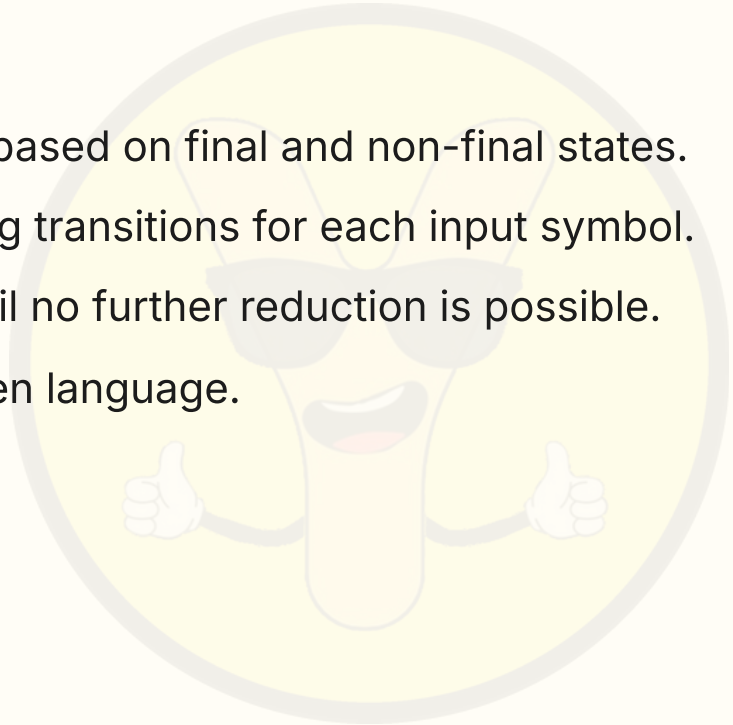
- A Moore machine can be converted to a Mealy machine by associating the state output with the incoming transitions.
- A Mealy machine can be converted to a Moore machine by splitting states based on their output values.
- Conversion preserves the input-output mapping behavior.



Minimization of Finite Automata

Reducing the number of states in a DFA without changing its language.

- Step 1: Remove unreachable states.
- Step 2: Partition states into groups based on final and non-final states.
- Step 3: Refine partitions by checking transitions for each input symbol.
- Step 4: Merge equivalent states until no further reduction is possible.
- Result: The minimal DFA for the given language.



Exam corner

- **Q1** Define DFA and NFA. Explain the difference between them with examples.
- **Q2** State and prove the equivalence of NFA and DFA using the subset construction method.
- **Q3** What is an ϵ-NFA? Explain the concept of ϵ-closure.
- **Q4** Differentiate between Moore and Mealy machines with a suitable example.
- **Q5** Explain the algorithm for the minimization of a DFA.
- **Q6** Convert a given NFA with ϵ to a DFA.
- **Q7** Define formal language and explain the hierarchy of languages.

Unit 1 done. You are exam-ready.

Revise the Exam corner once more, then open Unit 2 on theycotes.com. Every AKTU subject, unit by unit, free. Notes by Chirag Jain, Ycotes.

