ABHYASMITRA.IN
STUDY NOTES

THEORY OF COMPUTATIONS

Unit I — Introduction to Formal Languages and Finite Automata

Topics Covered (09 Hours)

Basic Mathematical Concepts: Sets, Symbols, and Strings
Formal Language Definition & Finite Representations
Language Operations: Union, Concatenation, Kleene Star & Plus
The Concept of Basic Machines (Automata Foundations)
Deterministic Finite Automata (DFA) Definition & Design
Nondeterministic Finite Automata (NFA) & Epsilon-NFA
Equivalence Conversions: E-NFA to NFA, NFA to DFA
DFA Minimization (Table-Filling & Partitioning Algorithms)
Finite Automata with Output: Moore and Mealy Machines
Inter-Conversion Between Moore and Mealy State Models

Compiled in a simple, easy-to-understand, book style format
for students of Computer Science and Engineering

TABLE OF CONTENTS

1.1 Basic Mathematical Concepts & Formal Languages

The Theory of Computation (TOC) is a branch of computer science that deals with how efficiently problems can be solved on a model of computation, using algorithms. Before we can design computing machines, we must establish a rigorous mathematical vocabulary for describing their inputs and outputs. This vocabulary is built upon sets, symbols, strings, and languages.

Sets, Symbols, and Alphabets

- **Sets:** A set is a collection of distinct objects. A set can be Finite (having a countable number of elements, e.g., the set of vowels V = {a, e, i, o, u}) or Infinite (having an endless number of elements, e.g., the set of all integers Z or real numbers R).
- **Symbols (Alphabets):** An alphabet (denoted by the capital Greek letter Sigma, Σ) is a finite, non-empty set of symbols. The elements of Σ are the basic building blocks of our strings. For example, Σ = {0, 1} is the binary alphabet, Σ = {a, b, ..., z} is the lower-case English alphabet, and Σ = {a, b} is a common two-symbol alphabet used in theoretical problems.

Strings & Substrings

A string (or word) is a finite sequence of symbols chosen from an alphabet Σ. For example, '0110' is a string over the alphabet Σ = {0, 1}.

- **Length of a String:** The length of a string w, denoted by |w|, is the number of symbols in the string. For example, if w = 'abc', then |w| = 3.
- **Empty String:** The empty string (denoted by epsilon, ε, or lambda, λ) is a string containing zero symbols. The length of the empty string is exactly 0, i.e., |ε| = 0.
- **Substring:** A substring is a sequence of consecutive symbols within a string. If w = 'automata', then 'auto' and 'mat' are substrings, while 'atm' is not (since the characters are not consecutive).
- **Concatenation:** If we have two strings x and y, their concatenation (written as xy) is the string formed by writing x followed by y. If x = 'dog' and y = 'house', then xy = 'doghouse'. The empty string acts as an identity element for concatenation: wε = εw = w.

Formal Language Definition

A Formal Language is a set of strings over a fixed alphabet. If Σ is our alphabet, we write Σ* to represent the set of all possible strings that can be formed using symbols from Σ, including the empty string ε. A language L is simply any subset of Σ* (L ⊆ Σ*). A language can be finite (containing a fixed list of words, e.g., L = {cat, dog}) or infinite (e.g., L = {a^n b^n | n ≥ 0}).

Operations on Languages

Since languages are mathematical sets, we can apply set operations to them, as well as string-specific operations:

- **Union (L1 ∪ L2):** The set of all strings that are in L1, or in L2, or in both.
- **Concatenation (L1 · L2):** The set of all strings formed by taking a string from L1 and concatenating it with a string from L2. Formally, L1L2 = {xy | x ∈ L1 and y ∈ L2}.
- **Kleene Star (L*):** Representing zero or more concatenations of strings from L. It always contains the empty string ε. L* = L^0 ∪ L^1 ∪ L^2 ∪ ... where L^0 = {ε}.
- **Kleene Plus (L+):** Representing one or more concatenations of strings from L. It does not contain ε unless ε is already in L. L+ = L^1 ∪ L^2 ∪ L^3 ∪ ...

Basic Machine Concept

A basic machine (or automaton) is an abstract mathematical model of a computing device. It takes an input string, transitions through a sequence of internal states based on the input symbols, and determines whether to accept or reject the string, or produces an output string.

1.2 Finite Automata Without Output

Finite Automata are abstract machines that have a finite number of internal states. They read an input string one symbol at a time, moving from state to state. We classify them based on whether their state transitions are deterministic or nondeterministic.

Deterministic Finite Automata (DFA)

In a DFA, for any given state and any input symbol, there is exactly one transition to a next state. There are no choices or ambiguities.

DFA Formal Definition (5-Tuple)

A Deterministic Finite Automaton is mathematically defined as a 5-tuple:

$M = (Q, \Sigma, \delta, q0, F)$

Where:

1. Q is a finite, non-empty set of internal States.

2. Σ is a finite set of input Symbols (the Alphabet).

3. δ is the Transition Function mapping $Q \times \Sigma \rightarrow Q$.

4. $q0 \in Q$ is the initial Start State.

5. $F \subseteq Q$ is the set of Final (Accepting) States.

Nondeterministic Finite Automata (NFA)

In an NFA, for a given state and input symbol, the machine can transition to zero, one, or multiple next states. This is modeled as a transition to a set of states (mapping $Q \times \Sigma \rightarrow P(Q)$, where $P(Q)$ is the power set of Q). The NFA accepts a string if there exists at least one valid path from the start state to any final state.

Epsilon-NFA (ε-NFA)

An extension of NFA that allows transitions without reading any input symbol, called ε-transitions. The machine can spontaneously change its state, which is useful for modular design of complex machines.

1.3 Equivalence Conversions & DFA Minimization

Although NFAs and ε-NFAs have different structural designs, they are computationally equivalent to DFAs. Any language accepted by an NFA can also be accepted by a DFA. We use specific algorithms to prove this equivalence.

NFA to DFA Conversion (Subset Construction)

To convert an NFA to a DFA, we represent each state of the DFA as a set of states of the NFA. If the NFA has N states, the resulting DFA can have up to 2^N states (though in practice, many states are unreachable).

Subset Construction Algorithm: Start by calculating the start state of the DFA, which is the set {q0}. For this set, and for each alphabet symbol 'a', compute the union of transitions from each state in the set. This forms a next state set. Repeat this for all newly discovered state sets until no new sets are found. Any state set containing a final state of the NFA becomes a final state of the DFA.

Epsilon-NFA to DFA Conversion

To convert an ε-NFA to a DFA, we must first define the Epsilon Closure (E-CLOSE) of a state. The E-CLOSE(q) is the set of all states reachable from q by taking zero or more ε-transitions. When performing subset construction, the start state of the DFA is E-CLOSE(q0). For each transition, we compute the E-CLOSE of the resulting state set.

Minimization of DFAs

For any regular language, there exists a unique DFA with the minimum possible number of states. Minimization algorithms identify and merge equivalent states. Two states p and q are equivalent if, for every input string w, the transitions from p and q both lead to accepting states or both lead to non-accepting states.

Minimization Algorithm (Partitioning): Divide the states into two groups: Group 1 (Final States) and Group 2 (Non-Final States). For each group, check if the states transition to the same group for each alphabet symbol. If a state transitions to a different group than the others, split it into a new subgroup. Repeat this process until the partitions stabilize. Merge the equivalent states within each subgroup to form the minimized DFA.

1.4 Finite Automata With Output: Moore and Mealy

Finite automata can also be designed to produce output. These machines do not have final states; instead, they generate a string of output symbols based on state transitions. The two primary models are Moore and Mealy machines.

Definitions and Structure

- **Moore Machine:** The output is associated entirely with the current state. The machine produces an output symbol immediately upon entering a state. If the input string has length L, the output string will have length L+1 (due to the initial output of the start state).
- **Mealy Machine:** The output is associated with the transition between states, depending on both the current state and the input symbol. If the input string has length L, the output string has length L.

Moore vs. Mealy Structural Comparison

Feature	Moore Machine	Mealy Machine
Output Dependability	Depends only on the current State.	Depends on both the current State and Input Symbol.
Output Position	Placed inside the state nodes.	Placed on the transition arcs.
Output String Length	L + 1 (includes initial state output).	L (matches input length exactly).
State Complexity	Requires more states to represent functions.	Requires fewer states to represent functions.

Inter-Conversion Algorithms

Moore and Mealy machines are computationally equivalent. We can convert any Moore machine into a Mealy machine, and vice versa:

- **Moore to Mealy Conversion:** For every state transition from state qi to qj with input symbol 'a', look at the output associated with state qj in the Moore machine. In the new Mealy machine, assign that output to the transition arc itself.
- **Mealy to Moore Conversion:** If a state q in the Mealy machine receives transitions that produce different outputs (e.g., one transition produces output '0' and another produces '1'), split state q into multiple states (q_0 and q_1) in the new Moore machine. Assign output '0' to q_0 and '1' to q_1, and redirect the transitions accordingly.

Unit Summary

This unit introduced the foundations of formal languages and finite automata.

- Formal languages are sets of strings built from alphabets. Operations include union, concatenation, and Kleene closures.
- Finite Automata without output (DFA, NFA, ε-NFA) are equivalent in power and accept regular languages.
- Conversions are performed using subset construction and epsilon closure calculations.
- DFA Minimization merges equivalent states using the partitioning algorithm to construct the unique minimal machine.
- Finite Automata with output include Moore machines (state-based output) and Mealy machines (transition-based output).