TAFL Lecture Notes

By
Dr. Bindeshwar Singh Kushwaha
CSE Department,
Ambalika Institute of Management and Technology (AIMT), Lucknow

Alphabet

Alphabet (Σ)

- An alphabet is a finite, non-empty set of symbols.
- Symbols are basic elements used to form strings.

Example:

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

Strings (Words)

String (Word)

- A string is a finite sequence of symbols from Σ.
- Strings are usually denoted by w.
- Length of a string is denoted by $|w|$.

Examples over $\Sigma = \{a, b\}$:

$$a, \quad b, \quad ab, \quad baab$$

$$w = baab, \quad |w| = 4$$

$$\varepsilon = \text{empty string}, \quad |\varepsilon| = 0$$

Language

Language

- A language is a collection (set) of strings formed from an alphabet.
- Each string in a language is made using symbols of the given alphabet.
- A language may contain a finite or infinite number of strings.

Example: If $\Sigma = \{a, b\}$

$$L = \{aa,\ abaa,\ baba,\ aabb,\ \ldots\}$$

- The above language contains strings having an even number of a.

Automaton

Automaton (Plural: Automata)

- An automaton is a mathematical model of computation.
- It is an abstract machine that processes input strings.
- It reads symbols one by one from an input alphabet.
- After processing the input, it either **accepts** or **rejects** the string.

Purpose:

- To recognize patterns
- To define formal languages

Components of an Automaton

An automaton consists of:

- **States (Q)** – Finite set of states
- **Alphabet (Σ)** – Set of input symbols
- **Transition Function (δ)** – Rules for moving between states
- **Start State (q_0)** – Initial state
- **Final State(s) (F)** – Accepting states

General Representation:

$$M = (Q, \Sigma, \delta, q_0, F)$$

Working of an Automaton

- The automaton starts in the start state q_0.
- It reads the input string symbol by symbol.
- For each symbol, it changes state according to the transition function.
- After reading the complete string:
  - If it is in a final state $\rightarrow$ String is **Accepted**
  - Otherwise $\rightarrow$ String is **Rejected**

Construction of Minimal DFA-Example-1

Problem: Construct a minimal DFA over $\{a, b\}$ that accepts all strings starting with a.

Step 1: Language Description

$$L = \{w \mid w \text{ starts with } a\}$$

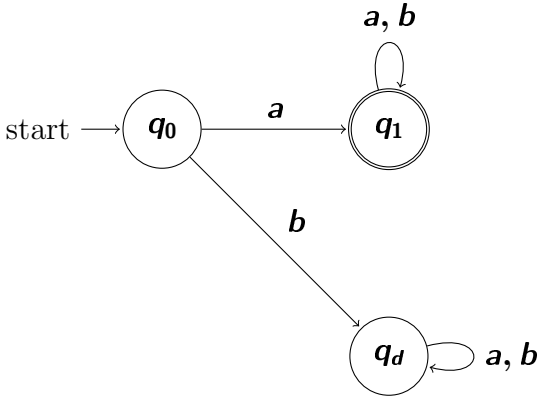
Accepted: a, ab, aa, aba
Rejected: b, ba, bb, ε

Step 2: Idea of Construction

- If first symbol is $a \rightarrow$ go to accepting state.
- If first symbol is $b \rightarrow$ go to dead state.
- After reading first a, any symbol is allowed.

Transition Diagram of the DFA

Language: All strings over $\{a, b\}$ starting with a



Construction of Minimal DFA-Example-2

Language: All strings over the alphabet $\{a, b\}$ that start with the symbol b.

Examples:

- Accepted: b, ba, bab, bba, $bbbb$, $\ldots$
- Rejected: a, aa, ab, ε (empty string)

Step 2: Idea of Construction

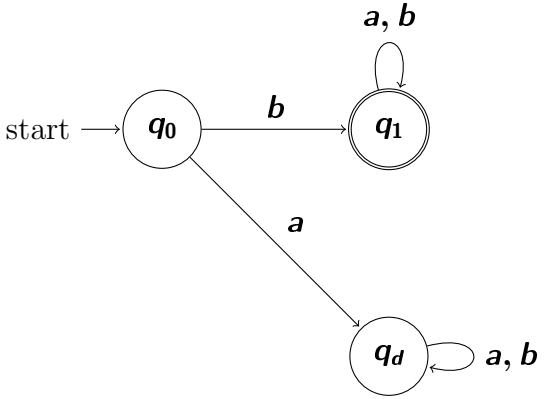
Idea of Construction:

- Start at q_0, the initial state.
- If the first symbol is b, go to the accepting state q_1.
- If the first symbol is a, go to the dead state q_d.
- After reaching q_1, any sequence of a and b keeps the automaton in q_1.
- The dead state q_d loops on all inputs to reject strings starting with a.

Step 3: Transition Rules

Present State	Input = a	Input = b
q_0	q_d	q_1
q_1	q_1	q_1
q_d	q_d	q_d

Step 4: Minimal DFA Diagram



Step 5: 5-Tuple Representation of DFA

The DFA can be formally represented as a 5-tuple:

$$M = (Q, \Sigma, \delta, q_0, F)$$

Where:

$$Q = \{q_0, q_1, q_d\}, \quad \Sigma = \{a, b\}, \quad q_0 \text{ is the start state}, \quad F = \{q_1\}$$

Transition function δ:

$$\delta(q_0, a) = q_d, \quad \delta(q_0, b) = q_1$$
$$\delta(q_1, a) = q_1, \quad \delta(q_1, b) = q_1$$
$$\delta(q_d, a) = q_d, \quad \delta(q_d, b) = q_d$$

Construction of Minimal DFA-Example-3

Language: All strings over the alphabet $\{a, b\}$ that start with the sequence `ab`.

Examples:

- Accepted: ab, aba, abb, $abba$, $abbab$, $\ldots$
- Rejected: a, b, aa, ba, bb, ε (empty string)

Step 2: Idea of Construction

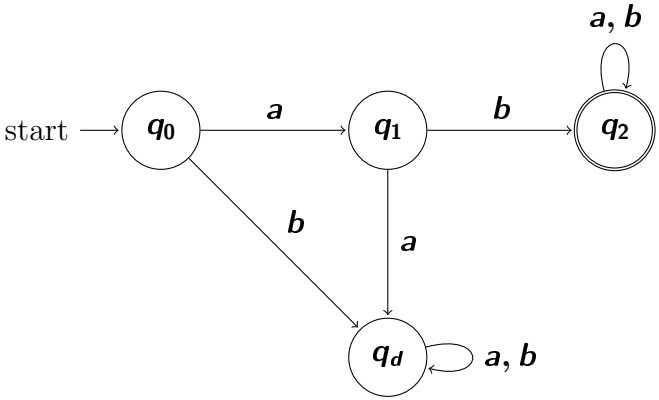
Idea of Construction:

- Start at q_0, the initial state.
- Read the first symbol:
  - If a, move to q_1 (expecting b next).
  - If b, go to dead state q_d.
- Read the second symbol:
  - If b, move to accepting state q_2.
  - If a, go to dead state q_d.
- From q_2, any sequence of a and b stays in q_2 (already accepted 'ab' prefix).
- Dead state q_d loops on all inputs.

Step 3: Transition Table

Present State	Input = a	Input = b
q_0	q_1	q_d
q_1	q_d	q_2
q_2	q_2	q_2
q_d	q_d	q_d

Step 4: Minimal DFA Diagram



Step 5: 5-Tuple Representation of DFA

The DFA can be formally represented as a 5-tuple:

$$M = (Q, \Sigma, \delta, q_0, F)$$

Where:

$$Q = \{q_0, q_1, q_2, q_d\}, \quad \Sigma = \{a, b\}, \quad q_0 \text{ is the start state}, \quad F = \{q_2\}$$

Transition function δ:

$$\delta(q_0, a) = q_1, \quad \delta(q_0, b) = q_d$$
$$\delta(q_1, a) = q_d, \quad \delta(q_1, b) = q_2$$
$$\delta(q_2, a) = q_2, \quad \delta(q_2, b) = q_2$$
$$\delta(q_d, a) = q_d, \quad \delta(q_d, b) = q_d$$

Construction of Minimal DFA-Example-4

Language: All strings over the alphabet $\{a, b\}$ that **end with 'a'**.

Examples:

- Accepted: a, ba, aba, bba, $abba$, $\ldots$
- Rejected: b, bb, $ab\ b$, ϵ (empty string)

Step 2: Idea of Construction

Idea of Construction:

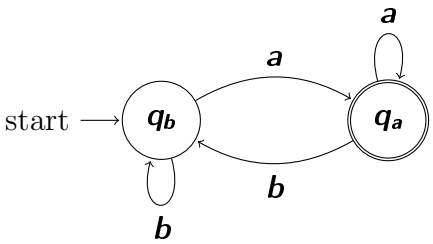
- Only the last symbol matters for acceptance.
- Define two states:
  - q_a → last symbol read was ‘a‘ → accepting
  - q_b → last symbol read was ‘b‘ → non-accepting
- Transitions:
  - On reading ‘a‘ → move to q_a
  - On reading ‘b‘ → move to q_b
- Start state can be q_b (empty string is not accepted).

Step 3: Transition Table

Present State	Input = a	Input = b
q_a	q_a	q_b
q_b	q_a	q_b

- $q_a \rightarrow$ accepting - $q_b \rightarrow$ non-accepting

Step 4: Minimal DFA Diagram



Step 5: 5-Tuple Representation of DFA

The DFA can be formally represented as a 5-tuple:

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

Where:

$Q = \{q_a, q_b\}$, $\Sigma = \{a, b\}$, q_b is the start state, $F = \{q_a\}$ strings end with 'a'

Transition function δ:

$$\delta(q_b, a) = q_a, \quad \delta(q_b, b) = q_b$$
$$\delta(q_a, a) = q_a, \quad \delta(q_a, b) = q_b$$

Construction of Minimal DFA-Example-5

Language: All strings over the alphabet $\{a, b\}$ that start with 'a' and end with 'b'.

Examples:

- Accepted: ab, aab, abb, $aaab$, $abab$, $\ldots$
- Rejected: b, ba, aba, a, ε

Step 2: Idea of Construction

Idea of Construction:

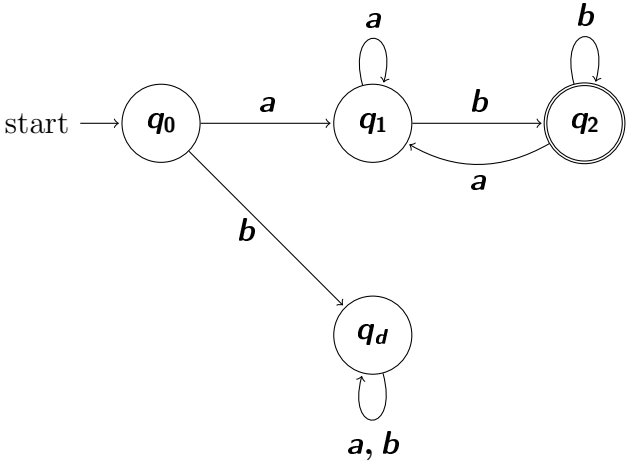
- The string must begin with ‘a‘.
- After the first symbol, we only need to track the **last symbol**.
- Define states:
  - q_0 → Start state
  - q_1 → Valid start, last symbol = ‘a‘
  - q_2 → Valid start, last symbol = ‘b‘ (accepting)
  - q_d → Dead state (string started with ‘b‘)

Step 3: Transition Table

Present State	Input = a	Input = b
q_0	q_1	q_d
q_1	q_1	q_2
q_2	q_1	q_2
q_d	q_d	q_d

- $q_2 \rightarrow$ accepting
- $q_0, q_1, q_d \rightarrow$ non-accepting

Step 4: Minimal DFA Diagram



Step 5: 5-Tuple Representation of DFA

The DFA can be formally represented as:

$$M = (Q, \Sigma, \delta, q_0, F)$$

Where:

$$Q = \{q_0, q_1, q_2, q_d\}$$
$$\Sigma = \{a, b\}$$
$$F = \{q_2\}$$

Transition function δ:

$$\delta(q_0, a) = q_1, \quad \delta(q_0, b) = q_d$$
$$\delta(q_1, a) = q_1, \quad \delta(q_1, b) = q_2$$
$$\delta(q_2, a) = q_1, \quad \delta(q_2, b) = q_2$$
$$\delta(q_d, a) = q_d, \quad \delta(q_d, b) = q_d$$

Construction of Minimal DFA-Example

Language: All strings over the alphabet $\{a, b\}$ that start with 'a' and end with 'a'.

Examples:

- Accepted: a, aa, aba, $abba$, $aabaa$, $\ldots$
- Rejected: b, ab, baa, abb, ε

Step 2: Idea of Construction

Idea of Construction:

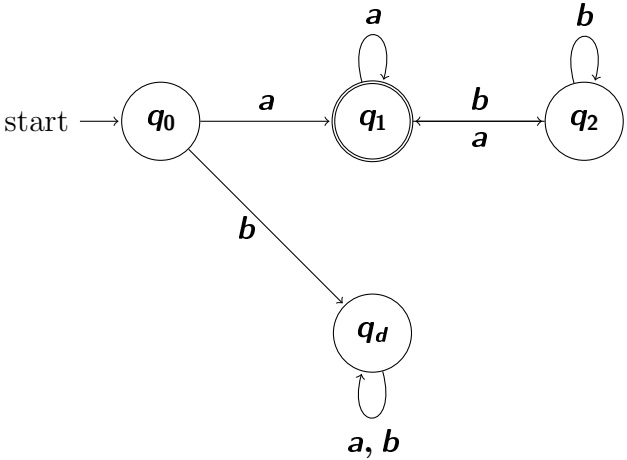
- The string must begin with ‘a‘.
- After the first symbol, we only need to track the **last symbol**.
- Define states:
  - q_0 → Start state
  - q_1 → Valid start, last symbol = ‘a‘ (accepting)
  - q_2 → Valid start, last symbol = ‘b‘
  - q_d → Dead state (string started with ‘b‘)

Step 3: Transition Table

Present State	Input = a	Input = b
q_0	q_1	q_d
q_1	q_1	q_2
q_2	q_1	q_2
q_d	q_d	q_d

- $q_1 \rightarrow$ accepting
- $q_0, q_2, q_d \rightarrow$ non-accepting

Step 4: Minimal DFA Diagram



Step 5: 5-Tuple Representation of DFA

The DFA can be formally represented as:

$$M = (Q, \Sigma, \delta, q_0, F)$$

Where:

$$Q = \{q_0, q_1, q_2, q_d\}$$
$$\Sigma = \{a, b\}$$
$$F = \{q_1\}$$

Transition function δ:

$$\delta(q_0, a) = q_1, \quad \delta(q_0, b) = q_d$$
$$\delta(q_1, a) = q_1, \quad \delta(q_1, b) = q_2$$
$$\delta(q_2, a) = q_1, \quad \delta(q_2, b) = q_2$$
$$\delta(q_d, a) = q_d, \quad \delta(q_d, b) = q_d$$

Q1: What is an Alphabet?

Answer:

- An alphabet ($\boldsymbol{\Sigma}$) is a finite, non-empty set of symbols.
- Symbols are basic building blocks used to form strings.

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

Q2: What is a String?

Answer:

- A string is a finite sequence of symbols from Σ.
- Length of string w is $|w|$.

Example:

$$w = baab, \quad |w| = 4$$

$$\varepsilon \text{ is empty string}, \quad |\varepsilon| = 0$$

Q3: What is a Language?

Answer:

- A language is a set of strings formed from an alphabet.
- It may be finite or infinite.

$$L = \{aa, abaa, baba, aabb, \dots\}$$

Q4: What is an Automaton?

Answer:

- Mathematical model of computation.
- Reads input symbol by symbol.
- Accepts or rejects string.

Q5: Components of Automaton

$$M = (Q, \Sigma, \delta, q_0, F)$$

- Q – States
- Σ – Alphabet
- δ – Transition function
- q_0 – Start state
- F – Final states

Q6: Working of Automaton

- Start at q_0
- Read symbols one by one
- Move using δ
- If final state $\in F \rightarrow$ Accept

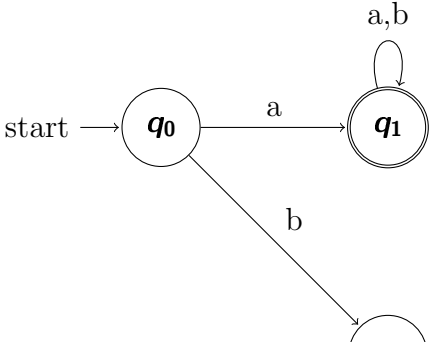
Q7: DFA – Strings Starting with a

Language:

$$L = \{w \mid w \text{ starts with } a\}$$

Idea:

- $a \rightarrow$ Accepting state
- $b \rightarrow$ Dead state

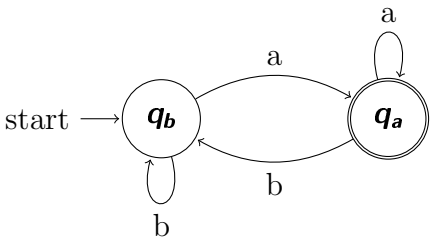


Q8: DFA – Strings Ending with a

Language:

$$L = \{w \mid w \text{ ends with } a\}$$

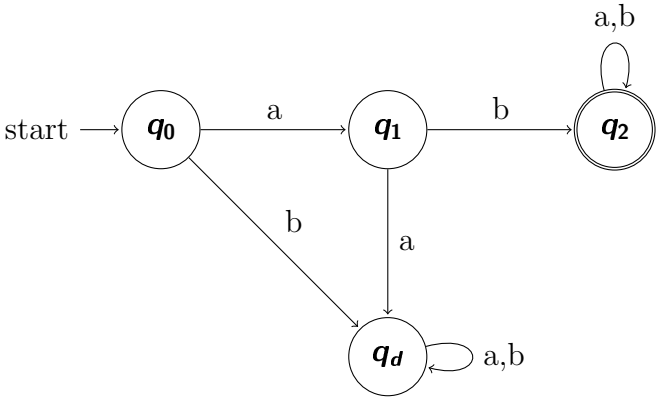
Idea: Track last symbol



Q9: DFA – Strings Starting with ab

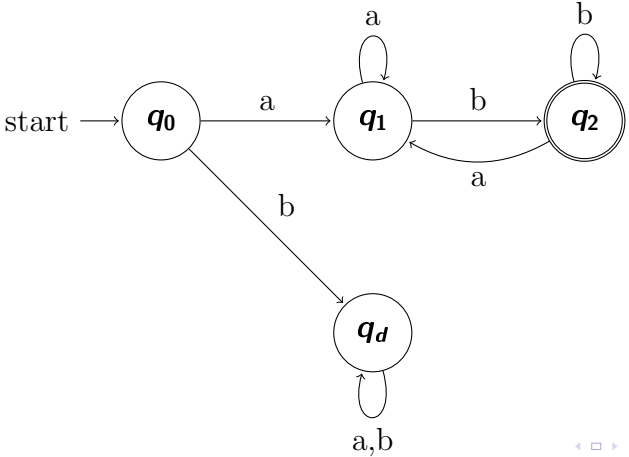
Language:

$$L = \{w \mid w \text{ starts with } ab\}$$



Q10: DFA – Start with a and End with b

$L = \{w \mid w \text{ starts with } a \text{ and ends with } b\}$



Q11: DFA – Start with a and End with a

$L = \{w \mid w \text{ starts with } a \text{ and ends with } a\}$