THEORY OF COMPUTATION

Decidability of $\texttt{ALL}_{\texttt{DFA}}$

Sunil Kumar (12342170)
Praveen Kumar Verma(12341630)

Problem Statement

Let $\texttt{ALL}_{\texttt{DFA}}$ be the language:

$$\texttt{ALL}_{\texttt{DFA}} = \{ \langle A \rangle \mid \text{A}isaDFAand\text{L(A)} = \Sigma^* \}$$

We want to prove that $\texttt{ALL}_{\texttt{DFA}}$ is **decidable**.

What Does Decidable Mean?

A language L is **Turing-decidable** if there exists a Turing Machine M (called a decider) such that:

- M accepts all strings in L,
- M rejects all strings not in L,
- M halts on all inputs (i.e., it never loops).

Key Concepts Involved

- **DFA Complementation:** For any DFA A, we can construct a new DFA B such that $L(B) = \Sigma^* - L(A)$ by swapping accepting and rejecting states.
- **Complement of Σ^*:** If $L(A) = \Sigma^*$, then its complement is $\Sigma^* - \Sigma^* = \emptyset$.
- **Emptiness Checking:** There exists a decider TM (from Theorem 4.4) that checks whether $L(B) = \emptyset$ for a given DFA B.

Constructing the Complement of a DFA A

To construct the complement of a DFA $A = (Q, \Sigma, \delta, q_0, F)$, we follow these steps:

1. **States:** The set of states of the complement DFA A' is the same as the set of states of A, i.e., $Q' = Q$.

2. **Alphabet:** The alphabet Σ' of the complement DFA A' is the same as the alphabet Σ of A, i.e., $\Sigma' = \Sigma$.

3. **Transition Function:** The transition function δ' of the complement DFA A' is the same as the transition function δ of A, i.e., $\delta'(q, a) = \delta(q, a)$ for all $q \in Q$ and $a \in \Sigma$.

4. **Start State:** The start state of the complement DFA A' is the same as the start state of A, i.e., $q_0' = q_0$.

5. **Accepting States:** The accepting states of A' are the non-accepting states of A, i.e., $F' = Q - F$.

This construction ensures that A' accepts exactly those strings that A does not accept, i.e., $L(A') = \Sigma^* - L(A)$.

High-Level Idea

We want to construct a Turing Machine M that decides $\texttt{ALL}_{\texttt{DFA}}$. To determine whether DFA A accepts all strings (i.e., $L(A) = \Sigma^*$), we:

1. Construct DFA B, the complement of A.

2. Use a TM T to check whether $L(B) = \emptyset$.

3. If so, accept (because that means $L(A) = \Sigma^*$), otherwise reject.

Formal Description of the Decider TM M

Input: $\langle A \rangle$, where A is a DFA.

Procedure:

1. Construct DFA B by swapping accepting and non-accepting states of A.

2. Run a decider TM T (from Theorem 4.4) on input $\langle B \rangle$ to check if $L(B) = \emptyset$.

3. If T accepts (i.e., $L(B) = \emptyset$), then accept $\langle A \rangle$.

4. Otherwise, reject $\langle A \rangle$.

Description of the Emptiness Decider TM T

Given a DFA $B = (Q, \Sigma, \delta, q_0, F)$, the Turing Machine T decides whether $L(B) = \emptyset$ as follows:

1. Perform a Breadth-First Search (BFS) starting from the start state q_0.

2. For each state visited, follow all transitions defined by δ.

3. If any accepting state $f \in F$ is reachable, then $L(B) \neq \emptyset$, so reject.
4. If no accepting state is reachable after the search, then $L(B) = \emptyset$, so accept.

This algorithm always halts and gives a correct yes/no answer.

Why This Works

- If $L(A) = \Sigma^*$, then $L(B) = \emptyset$, so T will accept and M accepts.
- If $L(A) \neq \Sigma^*$, then B will accept some string, so T rejects and M rejects.
- All steps are mechanical, finite, and guaranteed to halt.

Conclusion

The machine M decides the language $\mathtt{ALL}_{\mathtt{DFA}}$. Therefore,

- $\mathtt{ALL}_{\mathtt{DFA}}$ is **recursive**,
- That means it is **decidable**.