Decision Properties of Regular Languages

Sara Sorahi

Introduction to the Theory of Computation

What Is a Decision Property?

A **decision property** is a yes/no question about a language or an automaton.

Examples of decision questions:

- Is this string accepted by the automaton?
- Is the language empty?
- Is the language infinite?
- Do two automata accept the same language?

For regular languages, all major decision properties are **decidable**.

Membership Problem

Problem: Given a DFA M and a string w, decide whether $w \in L(M)$.

Key idea: A DFA has exactly one computation path for each input string. So we simply simulate the machine.

Membership: Concrete Example

Language:

$$L = \{\, w \in \{0,1\}^* \mid w \text{ has an even number of 1s} \,\}$$

Test the string:

$$w = 1011$$

Step-by-step:

- read 1 $\rightarrow$ odd
- read 0 $\rightarrow$ odd
- read 1 $\rightarrow$ even
- read 1 $\rightarrow$ odd

Final state is non-accepting $\Rightarrow w \notin L$.

Emptiness Problem

Problem: Given a DFA M, is $L(M) = \emptyset$?

Key observation: $L(M)$ is nonempty iff *some accepting state is reachable* from the start state.

Emptiness: Algorithm

Algorithm:

- View the DFA as a directed graph
- Perform BFS or DFS from the start state
- Check whether an accepting state is reachable

If no accepting state is reachable, the language is empty.

Emptiness: Example

Suppose a DFA has:

- one accepting state
- but that state is not reachable from the start state

Then:

$$L(M) = \emptyset$$

No input string can ever reach an accepting state.

Finiteness vs. Infiniteness

Problem: Is the regular language $L(M)$ finite or infinite?

Key insight: $L(M)$ is infinite iff:

- there is a cycle reachable from the start state, and
- from that cycle, an accepting state is reachable

Why Cycles Create Infinite Languages

If a cycle lies on a path to an accepting state:

- the cycle can be repeated arbitrarily many times
- each repetition produces a different string

This guarantees infinitely many accepted strings.

Finiteness: Example

Language:

$$L = \{\, w \in \{0,1\}^* \mid w \text{ has length } \leq 3 \,\}$$

Reason:

- no cycles on paths to accepting states
- only finitely many strings of length ≤ 3

Therefore, L is finite.

Equivalence Problem

Problem: Given two DFAs M_1 and M_2, do they accept the same language?

Formally:

$$L(M_1) = L(M_2) \ ?$$

Equivalence: Core Idea

Two languages are equal iff their symmetric difference is empty:

$$L_1 = L_2 \iff (L_1 \setminus L_2) \cup (L_2 \setminus L_1) = \emptyset$$

We reduce equivalence to the emptiness problem.

Equivalence: Example

L_1: strings with an even number of 1s
L_2: strings whose number of 1s is divisible by 2

These descriptions look different, but:

$$L_1 = L_2$$

Equivalence is about languages, not descriptions.

Containment Problem

Problem: Is $L_1 \subseteq L_2$?

Key equivalence:

$$L_1 \subseteq L_2 \iff L_1 \cap \overline{L_2} = \emptyset$$

Containment: Example

L_1: strings ending with 01
L_2: strings containing at least one 1

Every string ending with 01 contains a 1.

$$L_1 \subseteq L_2$$

Containment follows from logical implication between properties.

Membership: Another Example

Language:

$$L = \{\, w \in \{a, b\}^* \mid w \text{ ends with } ab \,\}$$

Test the string:

$$w = aab$$

Reasoning:

- last symbol is $\mathtt{b}$
- second-to-last symbol is $\mathtt{a}$

Therefore, $w \in L$.

Membership depends only on the last two symbols.

Emptiness: Proof with a Language

Language:

$$L = \{\, w \in \{0,1\}^* \mid w \text{ starts with 1 and ends with 0} \,\}$$

Claim:

$$L \neq \emptyset$$

Proof:

- Consider the string 10
- It starts with 1 and ends with 0
- Hence, $10 \in L$

Exhibiting one accepted string proves non-emptiness.

Infiniteness: Formal Proof Example

Language:

$$L = \{\, w \in \{0,1\}^* \mid w \text{ contains at least one } 1 \,\}$$

Claim:

$$L \text{ is infinite.}$$

Proof:

- The string $1 \in L$
- For any $n \geq 1$, the string $0^n1 \in L$
- All strings 0^n1 are distinct

Thus, L contains infinitely many strings.

Finiteness: Another Proof Example

Language:

$$L = \{ w \in \{a, b\}^* \mid |w| = 2 \}$$

Claim:

L is finite.

Proof:

- All strings have exactly length 2
- Possible strings: aa, ab, ba, bb
- There are exactly 4 such strings

A language with a fixed maximum length is always finite.

Equivalence: Proof Using Difference

Languages:

- L_1: strings over $\{0, 1\}$ with an even number of 1s
- L_2: strings where the number of 1s is divisible by 2

Claim:

$$L_1 = L_2$$

Proof:

- For any string, having an even number of 1s means divisibility by 2
- Thus, $L_1 \subseteq L_2$ and $L_2 \subseteq L_1$

Mutual containment implies equivalence.

Containment: Formal Proof

Languages:

- L_1: strings ending with 01
- L_2: strings containing at least one 1

Claim:

$$L_1 \subseteq L_2$$

Proof:

- Every string ending with 01 contains a 1
- Hence, no string in L_1 lies outside L_2

Containment follows from logical implication of properties.