Decision Problems

- We need a mechanism to describe a regular language in finite way.
  - DFA, NFA, \( \epsilon \)-NFA, or regular expression

- We have shown that any language expressed via one of these formalisms may be expressed via any of them.

- These proofs are constructive; i.e., there is an algorithm to compute the construction.

\[ \text{Claim: There is an algorithm to decide whether a given regular language is empty.} \]

\[ \text{Proof sketch:} \]

1. Convert to a machine if necessary.

2. Decide whether a final state is reachable from the start state (using, e.g., depth-first search).
**Claim:** There is an algorithm to decide whether a given regular language includes a given string.

**Proof sketch:**

1. Convert to a DFA if necessary.
2. Simulate the DFA on the string.

---

**Claim:** There is an algorithm to decide, given regular languages \( L_1 \) and \( L_2 \), whether \( L_1 \subseteq L_2 \).

**Proof sketch:**

1. Convert to DFAs if necessary.
2. Construct DFA accepting \( L_1 - L_2 = L_1 \cap \overline{L_2} \) \( = \overline{L_1} \cup L_2 \).
3. Decide whether the language accepted by this DFA is empty.
Claim: There is an algorithm to decide, for given regular languages $L_1$ and $L_2$, whether $L_1 = L_2$.

Proof: Decide

1. whether $L_1 \subseteq L_2$; and

2. whether $L_2 \subseteq L_1$. 