5. Decidablity

Decidable problems concerning regular languages

1️⃣ Let $A_{DFA} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$. Then for every $w \in \Sigma^*$ and DFA B, we have

$$w \in L(B) \iff \langle B, w\rangle \in A_{DFA}$$

Theorem. A_{DFA} is a decidable language.

Proof. (sketch-up) Check whether the input w is valid; if not, reject. Then simulate B on input w. If the simulation ends in an accpeting state, then accept. If it ends in a nonaccepting state, then reject.

2️⃣ Let $A_{NFA} = \{\langle B, w\rangle \mid B \text{ is a NFA that accepts input string } w\}$. Then for every $w \in \Sigma^*$, we have

$$w \in L(B) \iff \langle B, w\rangle \in A_{NFA}$$

Theorem. A_{NFA} is a decidable language.

Proof. (sketch-up) Check whether the input w is valid; if not, reject. Then construct an equivalent DFA according to the NFA B, then we can directly call the previous procedure to check.

3️⃣ Let $A_{REX} = \{\langle R, w\rangle \mid R \text{ is a regular expression that generates string } w\}$.

Theorem. A_{REX} is a decidable language.

Proof. (sketch-up) Check whether the input w is valid; if not, reject. Convert regular expression R to an equivalent NFA, then we can directly call the procedure to check.

4️⃣ Let $E_{DFA} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \varnothing\}$

Theorem. E_{DFA} is a decidable language.

Proof. (sketch-up) Use the conclusion in chapter 1, we can use DFS to check the answer.

5️⃣ Let $EQ_{DFA} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}$

Theorem. EQ_{DFA} is a decidable language.

Proof. (sketch-up) Use the conclusion in chapter 1, we have

$$L(A) = L(B) \iff L(C) = \Big(L(A) \cap \overline{L(B)}\Big) \cup \Big(\overline{L(A)} \cap L(B)\Big) = \varnothing$$

Then we construct a DFA C to recognize $L(C)$, then call the previous procedure to check whether $L(C) = \varnothing$.

Decidable problems concerning context-free languages

1️⃣ Let $A_{CFG} = \{\langle G, w\rangle \mid G \text{ is a CFG that generate } w\}$.

Theorem. A_{CFG} is a decidable language.

Proof. (sketch-up) Use Chomsky normal form, and then according to the property, any derivation of w has $2|w| - 1$ steps. We then can enumerate all situations to decide.

2️⃣ Let $E_{CFG} = \{\langle G \rangle \mid G \text{ is a CFG and } L(G) = \varnothing\}$.

Theorem. E_{CFG} is a decidable language.

Proof. (sketch-up)

- Mark the terminals;
- If $A \to B_1 B_2 \cdots B_k$ and all B_i are marked, then mark A;
- Repeat the process until no more variable can be marked;
- Check whether the start variable is marked. If so then reject; otherwise, accept.

3️⃣ Let $EQ_{CFG} = \{\langle G, H \rangle \ \mid G \text{ and } H \text{ and CFGs and } L(G) = L(H)\}$.

Theorem. EQ_{CFG} is not a decidable language.

4️⃣

Theorem. Every context-free language is decidable.

Proof. (sketch-up) Construct a PDA that recognize the language, then convert it to CFG. Use the previous procedure to check.

Relationship among classes of languages

$$Regular \subset Context - free \subset Decidable \subset Turing - recognizable$$

Undecidablity

Testing membership: $A_{TM} = \{\langle M, w \rangle \mid M \text{ is a TM and } M \text{ accept } w\}$

Theorem. A_{TM} Is Turing-recognizable.

Proof. U on $\langle M, w \rangle$:

1. Simulate M on w;
2. If M enters its accept state, then accept; else reject.

U is a universal Turing machine first proposed by Alan Turing in 1936. This machine is called universal because it is capable of simulating other Turing machine from the description of that machine.

Theorem. $\mathbb{R}$ is not countable. (use diagonalization method to prove)

Corollary. Some languages are not Turing-recognizable.

Proof. All languages that TM can recognize is countable, $|L^R| = |\mathbb{N}|$. (a TM can be viewed as a 01 string that has a fixed size, and the union of countable number of countable set is also countable) But, all languages is uncountable, $|L| = 2^{|\mathbb{N}|} = |\mathbb{R}|$. So there must be some languages that are not Turing-recognizable.

Theorem. A_{TM} Is undecidable.

Proof. Asuume H is a decider for A_{TM}. We construct D on $\langle M \rangle$, where M is a TM.

- Run H on input $\langle M, \langle M \rangle \rangle$.
- Output the opposite of what H outputs. If rejects, then accept; otherwise, reject.
- Then, consider $D(\langle D \rangle)$. $D(\langle D \rangle)$ accepts if D rejects $\langle D \rangle$; otherwise rejects; then we have derived a contradiction.

co-Turing-recognizable: A language is co-Turing-recognizable if it is the complement of a Turing-recognizable language.

Theorem. A language is decidable iff it is Turing-recognizable and co-Turing-recognizable.

Proof. ($\Longleftarrow$) Combined two machines together. ($\Longrightarrow$) Simple.

Corollary. $\overline{A_{TM}}$ is not Turing-recognizable.

Proof. Otherwise, it is decidable according to the previous theorem.)