1. Recognizable vs. Decidable Languages

A Turing Machine M is called a **recognizer** for a language L over the alphabet Σ if the following statement is true:

$$\forall w \in \Sigma^*.\ (w \in L \leftrightarrow M \text{ accepts } w)$$

A TM M is called a **decider** for L if the following two things are true:

$$M \text{ is a recognizer for } L, \text{ and also}$$
$$\forall w \in \Sigma^*.\ (M \text{ halts on } w)$$

A language L is called **recognizable** if there exists a recognizer for L, and it is called **decidable** if there exists a decider for L. Let's build some intuition with these definitions.

a. Say we run a TM M on an input w. M could accept, reject, or loop on w.

   (1) Which outcomes count as "halting"?

   (2) Which outcomes count as "not accepting"?

b. Say a TM M is a recognizer for the language L_1, and we run M on the input w.

   (1) If $w \in L_1$, what are the possible outcomes?

   (2) What about if $w \notin L_1$?

c. Say a TM M is a decider for the language L_2, and we run M on the input w.

   (1) If $w \in L_2$, what are the possible outcomes?

   (2) What about if $w \notin L_2$?

d. Are these statements true or false? Why or why not?

   (1) Every language is recognizable. In other words, given a language L, we can always find a TM M where M is a recognizer for L.

   (2) Every TM is a recognizer for some language. In other words, given a TM M, we can always find a language L where M is a recognizer for L."

   (3) Every decidable language is recognizable.

   (4) Every regular language is recognizable.

e. Say a TM M is **not** a decider for the language L_3.

   (1) What does this mean? (Hint: Take the negation of the definition of a decider.)

   (2) Knowing that M is not a decider for L_3, can we say whether or not L_3 is decidable? (Hint: Take the negation of the definition of decidable.)

f. Why is the set of decidable languages, **R**, a subset of the set of recognizable languages, **RE**?

2. Guide to CFGs

We'll walk through useful ideas from the Guide to CFGs. Write a CFG for these languages:

a. Build related quantities together

(1) $L = \{a^n b^{2n} \mid n \in \mathbb{N}\}$ with $\Sigma = \{a, b\}$

(2) $L = \{a^x b^y c^y d^x \mid x, y \in \mathbb{N}\}$ with $\Sigma = \{a, b, c, d\}$

b. Separate the possible outputs into cases or simpler concatenated strings

(1) $L = \{w \mid w$ is non-empty and consists of all a's or all b's$\}$ with $\Sigma = \{a, b\}$

(2) $L = \{w \mid w$ contains at least one a and at least one b$\}$ with $\Sigma = \{a, b\}$

(3) $L = \{a^x b^y c^z \mid x, y, z \in \mathbb{N}$ and $x = y$ or $x = z\}$ with $\Sigma = \{a, b, c\}$

c. Break down strings in the language recursively

(1) $L = \{a^n \mid n$ is odd$\}$ with $\Sigma = \{a\}$, using only one nonterminal

(2) L^* where $L = \{a^n b^n \mid n \in \mathbb{N}\}$ with $\Sigma = \{a, b\}$

3. Proofs on Languages

Let $f : \wp(\Sigma^*) \to \wp(\Sigma^*)$ be a function defined as $f(L) = \{ww \mid w \in L\}$.

a. Prove that f is injective. Hint: What do you know if two sets are not equal? How do you show that two sets are not equal?

Let $g : \wp(\Sigma^*) \to \wp(\Sigma^*)$ be a function defined as $g(L) = L^*$.

b. Prove that g is not surjective.

c. Let M be a monoid. Here is the definition of a monoid: $\varepsilon \in M$ and $MM \subseteq M$.

- Prove this statement: For any natural number n, $M^n \subseteq M$.
- Prove that $g(M) = M$.