Selected Q

Show that the language $\{0^m 1^n : m \neq n\}$ is not regular. (Hint: you may find that pumping theorem does not work well in this case. Try the closure property.)

Q5. Assume that $L_1 = \{0^m 1^n : m \neq n\}$ is regular. Since $L = L(0^*1^*)$ is regular, by the closure property of regular languages, $L - L_1 = \{0^n 1^n : n \geq 0\}$ is regular (because $L - L_1 = L \cap \overline{L}_1$). It is, however, known that $\{0^n 1^n : n \geq 0\}$ is not regular. Contradiction.

(a) If L is a non-empty finite language, then the minimum pumping length that works for L is 1+ the length of the longest string in L. √

(b) Let $G = (V, \Sigma, S, R)$ be some context-free grammar in Chomsky norm form. For any string $w \in L(G)$, the number of distinct derivations from S to w is finite. 对 2|w|-1

Q1. Prove that the following language is not recursive, but is recursively enumerable.

$$L_1 = \{\text{“}M\text{”} : M \text{ is a Turing machine that halts on at least 2023 strings.}\}$$

Q1. We first show L_1 is not recursive by reducing H to L_1. Suppose that some Turing machine M_1 decides L_1. Consider the following Turing machine M_H.

M_H = on input “M”“w”:

1.construct a Turing machine $\widetilde{M}$ as follows

$\widetilde{M}$ = on input x

1. run M on w

2. run M_1 on “$\widetilde{M}$” and the return the result

If M halts on w, then $\widetilde{M}$ halts on every input. If M does not halt on w, $\widetilde{M}$ halts on no input. Therefore, M halts on w if and only if $\widetilde{M}$ halts on at least 2023 strings (that is, M_1 accepts “$\widetilde{M}$”). M_H decides H. This completes the reduction.

Next we show that L_1 is recursive enumerable by presenting a Turing machine M_1' to semidecide it. We label the strings in Σ^* as $s_1, s_2, \ldots$ in incresing length.

构造一个图灵机来证明，采用了折现法来遍历

at least这样枚举的题目都是这样用折线来枚举

M_1' = on input “M”:

1.For $i = 2023, 2024, \ldots$

2. For $s = s_1, s_2, \ldots, s_i$

3. Run M on s for i steps

4. If M halts on at least 2023 strings

5. halt

证明r.e一般就是构造一个图灵机来半判定（如果还有w，就要构造广义图灵机），另一种小的方法是规约到H

Q3. Prove that the following language is not recursively enumerable. (Hint: you may reduce $\overline{H}$ to L_3.)

$L_3 = \{$"M" : M is a Turing machine such that there are at least 2023 strings on which M does not halt.$\}$

Q3. We already know that $\overline{H}$ is not recursively enumerable.
reduce H to L_3. Given any Turing machine M and its input w, we construct the following Tuing machine

$$f(\text{"}M\text{" "}w\text{"}) = \text{on input "}x\text{":}$$
$$1.\text{run } M \text{ on } w$$

If M halts on w, then $f(M, w)$ halts on every input. If M does not halt on w, $f(M, w)$ halts on no input. Therefore, M does not halt on w if and only if there are at least 2023 strings on which $f(M, w)$ does not halt. In otherwords, "M""w" $\in \overline{H}$ if and only if $f($"M" "w"$) \in L_3$. f is a reduction from $\overline{H}$ to L_3. This completes the proof.

Theory of Computation, Fall 2023
Quiz 3

Q1. Show that the following language is decidable. You may use any conclusion that we have proved in class.

$$S = \{\text{“}M\text{” is a DFA and } M \text{ accepts } w^R \text{ whenever it accepts } w\}$$

Q2. Prove that the following language is not recursive. You may reduce from any language that has been proved to be non-recursive in class.

$$A = \{\text{“}M_1\text{”“}M_2\text{”} : M_1 \text{ and } M_2 \text{ are two Turing machines with } L(M_1) \cap L(M_2) \neq \emptyset\}$$

Theory of Computation, Fall 2023
Quiz 3 Solutions

Q1. In class, we have proved that EQ_{DFA} is recursive. Suppose Turing machine M_{EQ} decides

$$EQ_{DFA} = \{"M_1""M_2" : M_1 \text{ and } M_2 \text{ are two DFAs with } L(M_1) = L(M_2)\}.$$

To prove that S is recursive, it suffices to reduce S to EQ_{DFA}. We construct a Turing machine M_S that decides S as follows.

$M_S =$ on input "M" :
1. construct a DFA M_R with $L(M_R) = \{w^R : w \in L(M)\}$
2. run M_{EQ} on "M""M_R"
3. return the result of M_{EQ}

This completes the proof.

Q2. Let $L = \{"M" : "M" \text{ is a Turing machine that halts on some input}\}$. In class, we have proved that L is not recursive. To prove that A is not recursive, it suffices to reduce L to A. Suppose there is a Turing machine M_A decides A. Then we can construct a Turing machine M_L that decides L as follows.

$M_L =$ on input "M" :
1. construct a Turing machine M_{all} that halts on every input
2. run M_A on "M""M_{all}"
3. return the result of M_A