Theory of Computation – Lecture 11
MARCH 8, 2021

- Rice’s Theorem
- The Recursion Theorem (Section 6.1)
- Measuring Complexity (Section 7.1)

Some material from slides by M. Sipser
Homework 3 assigned today.
- It is due Wednesday March 17, 2021 at 11:59pm (NY Time)
Review of Mapping Reducibility

- Define: $E_{TM} = \{ <M> | M \text{ is a TM and } L(M) = \emptyset \}$.
  - We showed $E_{TM}$ is undecidable.
- Define $EQ_{TM} = \{ <M_1, M_2> | M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2) \}$
  - Prove $EQ$ is undecidable by showing $E_{TM} \leq_m EQ_{TM}$. 
Rice's Theorem

- We have shown several undecidability results about languages.
  - $A = \{\langle M, w \rangle \mid M \text{ is a Turing machine and } M \text{ accepts } w \}$ is undecidable.
  - $H = \{\langle M, w \rangle \mid M \text{ is a TM and } M \text{ halts on input } w \}$ is undecidable.
  - $E_{\text{TM}} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset \}$ is undecidable.
  - $E_{\text{EQ}_{\text{TM}}} = \{\langle M_1, M_2 \rangle \mid M_1 \text{ and } M_1 \text{ are TMs and } L(M_1) = L(M_2) \}$ is undecidable.

- Rice's Theorem says that any non-trivial property of the language recognized by a TM is undecidable.
  - A set $P$ of languages is a non-trivial property of Turing-recognizable languages such that there is some TM $M$ such that $L(M) \in P$, and there is some TM $N$ such that $L(N) \notin P$.

- $P$ is a property of the TM’s language.
  - If 2 TMs have the same language, either both are in $P$ or both are not in $P$. 
Rice’s Theorem: Let $P$ be a nontrivial property. Then $M_P = \{ \langle M \rangle \mid L(M) \in P \}$ is undecidable.
Application of Rice’s Theorem

- $P = \text{languages that contain the string 01}$
- $M_P = \{\langle M \rangle \mid 01 \in L(M)\}$
The Recursion Theorem

SECTION 6.1
Recursive Programs

M: On input $w$
   
   If $w = \lambda$ then output 0
   
   Else
     
     Obtain $\langle M \rangle$
     
     Run $M$ on $\text{tail}(w)$
     
     If $M$ outputs a number $n$ then output $n+1$
**The Recursion Theorem**

- **Recursion theorem:** Let $T$ be a Turing Machine that computes a function $t : \Sigma^* \times \Sigma^* \rightarrow \Sigma^*$. There is a Turing Machine $R$ that computes a function $r : \Sigma^* \rightarrow \Sigma^*$, where for every $w$, $r(w) = t(<R>, w)$. 
Proof of Recursion Theorem

- **Lemma**: There is a computable function $q: \Sigma^* \to \Sigma^*$, where if $w$ is any string, $q(w)$ is the description of a TM that prints out $w$ and then halts.
Recursion theorem: Let $T$ be a Turing Machine that computes a function $t: \Sigma^* \times \Sigma^* \rightarrow \Sigma^*$. There is a Turing Machine $R$ that computes a function $r: \Sigma^* \rightarrow \Sigma^*$, where for every $w$, $r(w) = t(\langle R \rangle, w)$.
Application of Recursion Theorem (1)

- Prove $A = \{ \langle M, w \rangle \mid M$ is a TM and $M$ accepts $w$} is undecidable.
A TM is **minimal** if there is no equivalent TM with a shorter description.

\[ \text{MIN}_{\text{TM}} = \{ \langle M \rangle \mid M \text{ is a minimal TM} \} \]

**Theorem:** \( \text{MIN}_{\text{TM}} \) is not Turing-recognizable.
Time Complexity – Chapter 7
Intro to Time Complexity

- So far, we have focused on computability theory (1930s – 1950s)
  - Is language A decidable?
- Complexity theory (1960s – present)
  - Is language A decidable with restricted resources?
  - Resources = time, memory, ...
- We will start our study of complexity theory with time complexity
  - How many steps are needed to decide a language A?
    - A function of the input
  - We give an upper bound for all inputs of lengths n.
    - Called worst-case time complexity.
Number of steps to decide $A = \{ a^kb^k \mid k \geq 0 \}$

- $M =$ On input $w$
  1. Scan input to check if $w \in a^*b^*$. If not, reject.
  2. Repeat until all crossed off.
     - Scan tape, crossing off one $a$ and one $b$.
     - Reject if only $a$’s or only $b$’s remain.
  3. Accept if all crossed off.
Deciding $A = \{ a^k b^k \mid k \geq 0 \}$ faster

- **Theorem**: A 1-tape TM can decide $A$ using $O(n \log n)$ steps.
Deciding $A = \{ a^k b^k \mid k \geq 0 \}$ even faster

- **Theorem:** A multi-tape TM can decide $A$ using $O(n)$ steps.
Model Dependence

- Number of steps to decide $A = \{a^k b^k \mid k \geq 0\}$ depends on the model.
  - Single tape TM: $O(n \log n)$
  - Multi-tape TM: $O(n)$

- Computability theory: model independent
- Complexity theory: model dependent
  - For “reasonable” deterministic models, dependence is low (polynomial).
  - So, we will focus on single tape TM as basic model for complexity.