Lecture 10: Diagonalization, Undecidability, and Reductions

Date: September 21, 2023.

Recap: Recall the following concepts related to a Turing machine M.

- $\mathbf{L}(M) = \{x \mid M \text{ accepts } x\}$
- M is total if it halts on all inputs.
- M **recognizes/accepts** L if $\mathbf{L}(M) = L$. A language L is **recursively enumerable** (RE) if there is a TM M such that $L = \mathbf{L}(M)$.
- M **decides** L if $\mathbf{L}(M) = L$ and M is total. A language L is **decidable/recursive** (REC) if there is some TM M that decides L.
- Every object (graphs, programs, etc.) can be encoded as a binary string. The encoding of an object O will be denoted as $\langle O \rangle$.
  - Encoding of TM M is $\langle M \rangle$
  - For a binary string x, M_x denotes the TM M whose encoding is x, i.e., $\langle M \rangle = x$

Theorem 1. *The universal TM U recognizes the language* $\mathrm{MP} = \{\langle M \rangle \# \langle x \rangle \mid x \in \mathbf{L}(M)\} \in$ RE.

Computable Functions: A function $f : \Sigma^* \to \Sigma^*$ is **computable** if there is **total** TM M such that on input x, M halts with $f(x)$ written on its tape.

Reductions: For $A, B \subseteq \Sigma^*$, we say A **many-one reduces to** B (denoted $A \leq_m B$) if there is a *computable* function $f : \Sigma^* \to \Sigma^*$ such that

$$x \in A \text{ iff } f(x) \in B.$$

We often drop "many-one" and just say A reduces to B.

Important Decision Problems:

- $\mathrm{K} = \{\langle M \rangle \mid \langle M \rangle \in \mathbf{L}(M)\}$
- $\mathrm{MP} = \{\overset{\langle M\rangle \# x}{\cancel{\langle M, x \rangle}} \mid x \in \mathbf{L}(M)\}$
- $\mathrm{HP} = \{\overset{\langle M\rangle \# x}{\cancel{\langle M, x \rangle}} \mid M \text{ halts on } x\}$
- $\mathrm{EMPTY} = \{\langle M \rangle \mid \mathbf{L}(M) = \emptyset\}$
- $\mathrm{FIN} = \{\langle M \rangle \mid \mathbf{L}(M) \text{ is a finite set}\}$

Proposition 2. *If L is decidable/recursive then $\overline{L}$ is also decidable.*

TM : String $\to$ {true, false}

$\exists$ total TM M s.t $L(M) = L$.

Algorithm $\overline{M}$ for $\overline{L}$: Input x

Run M on x

If M accepts x

return No/false

else

return Yes/true.

If M that accepts x when $x \in L$ and does not halt when $x \notin L$.

$L(\overline{M}) = \phi$

$L(\overline{M}) = \overline{L}$.

Proposition 3. *L is decidable if and only if L and $\overline{L}$ are recursively enumerable.*

If L is REC then L is r.e. (follows from the defn)

($\Rightarrow$) L is decidable then L is r.e.

L is decidable $\Rightarrow$ $\overline{L}$ is decidable $\Rightarrow$ $\overline{L}$ is r.e.

($\Leftarrow$) $\exists$ TMs M and $\overline{M}$ s.t $L(M) = L$ and $L(\overline{M}) = \overline{L}$

Algo for L : Input x

For $i = 0$ to ∞.

Run M on x for i steps

Run $\overline{M}$ on x for i steps

if M accepts

return true/accept

if $\overline{M}$ accepts return false/reject

Input x

Run M on x

If M accepts then accept

Run $\overline{M}$ on x

If $\overline{M}$ accepts then reject

Proposition 4. $\overline{\mathrm{K}} = \{\langle M \rangle \mid \langle M \rangle \notin \mathbf{L}(M)\}$ *is not recursively enumerable.* (Diagonal Language)

$\overline{K} = \{ x \mid x \in L(M_x) \}$.

Suppose (for contradiction) TM M_y solves $\overline{K}$ i.e. $L(M_y) = \overline{K}$.

Case $y \in L(M_y) \Rightarrow y \notin \overline{K} \Rightarrow y \in L(M_y) \setminus \overline{K}$.

Case $y \notin L(M_y) \Rightarrow y \in \overline{K} \Rightarrow y \in \overline{K} \setminus L(M_y)$.

Contradiction

	ϵ	0	1	01	10	11	...
M_ϵ	$\boxed{N}$	N	N	N	...		
M_0	Y	$\boxed{Y}$	N	...			
M_1			$\boxed{}$				
M_{01}							
$\vdots$							

Proposition 5. *If $A \leq_m B$ and B is recursively enumerable (recursive) then A is recursively enumerable (recursive).*

$A \leq_M B \Rightarrow \exists f : \Sigma^* \to \Sigma^*$ s.t f is computable ($\exists N_f$ that computes f)

B is r.e (rec) $\Rightarrow \exists M$. (total) s.t $L(M) = B$.

Algo A : Input x.

Compute $f(x)$ (using N_f)

Run M on $f(x)$

If M accepts then accept

else reject.

$x \in A \Rightarrow f(x) \in B$

$\Rightarrow f(x)$ accepted by M

$\Rightarrow x$ is accepted.

$x \notin A \Rightarrow f(x) \notin B$

$\Rightarrow f(x)$ is not accepted by M

$\Rightarrow x$ is not accepted

If $A \leq_M B$ and A is not r.e (not rec) then B is not r.e (not rec)

Proposition 6. *If $A \leq_m B$ and $B \leq_m C$ then $A \leq_m C$.*

$\forall A \leq_M A$: through reduction identity fn.

$A \leq_M B$: $\exists f$ reduction.

$B \leq_M C$: $\exists g$ reduction.

What is the reduction from A to C? $g \circ f$.

Input x

Compute $f(x)$

Compute $g(f(x))$

$x \in A \Leftrightarrow f(x) \in B \Leftrightarrow g(f(x)) \in C$

Proposition 7. *If $A \leq_m B$ then $\overline{A} \leq_m \overline{B}$.*

$A \leq_M B$: $\exists$ reduction f.

Claim: f is a reduction from $\overline{A}$ to $\overline{B}$.

Problem 1. Prove that K $\leq_m$ MP. Is MP decidable?

$$\begin{array}{ccc} K & f & MP \\ \langle M\rangle & \rightsquigarrow & \langle M'\rangle \# x \end{array}$$

s.t. $\langle M\rangle \in K$ iff $\langle M'\rangle \# x \in MP$.

$\langle M\rangle \in L(M)$ iff $x \in L(M')$

$f(\langle M\rangle) = \langle M\rangle \# \langle M\rangle$.

$\Rightarrow \overline{K} \leq_m \overline{MP}$

$\overline{K} \notin RE \Rightarrow \overline{MP} \notin RE$.

MP is not decidable.

Problem 2. Prove that MP $\leq_m$ HP. Is HP decidable?

$$\begin{array}{ccc} MP & f & HP \\ \langle M\rangle \# x & \rightsquigarrow & \langle M'\rangle \# x' \end{array}$$

s.t. $x \in L(M)$ iff M' halts on x'.

M': Input w.

Run M on w

If M accepts

accept

else

go into infinite loop.

$f(\langle M\rangle \# x) = \langle M'\rangle \# x$

$x \in L(M) \Rightarrow M'$ will halt on x

$x \notin L(M) \Rightarrow M'$ will not halt on x.

Problem 3. Prove that $\overline{\text{K}} \leq_m$ EMPTY.

Problem 4. Prove that $\overline{\text{K}} \leq_m$ FIN and $\overline{\text{K}} \leq_m \overline{\text{FIN}}$.