Lec 16. Decidable and undecidable languages

Eunjung Kim

Unrecognizable languages

Concrete example of unrecognizable language

Let $A_{TM} = \{(M, w) : \ M \text{ is a TM and } M \text{ accepts } w\}$.

Then $\bar{A}_{TM} := \{0, 1\}^* \setminus A_{TM}$ is not Turing-recognizable.

Follows from the undecidability of A_{TM} and the characterization of undecidable languages.

Decision Problem

Membership test for a language A

Consider a language $A \subseteq \Sigma^*$.

INPUT: a string $w \in \Sigma^*$.

TASK: decide if $w \in A$ or not; that is, output YES ("accept") if $w \in A$, output NO ("reject") otherwise.

The language A itself is also called a decision problem.

Solving a decision problem A

Solving a (decision) problem A means having an algorithm for A, i.e. an algorithm for the membership test for A. By Church-Turing Thesis, this means to have a Turing machine M which decides A, i.e.

$$M(w) = \begin{cases} \text{ACCEPT} & \text{if } w \in A \\ \text{REJECT} & \text{otherwise.} \end{cases}$$

Examples of Decision Problems

- Decide if a given context-free grammar G generates a given string w: corresponds to a membership test for the language

$$\{\langle G, w\rangle \mid G \text{ is a CFG that generates } w\}.$$

- Decide if a graph is connected: corresponds to a membership test for the language

$$\{\langle G\rangle \mid G \text{ is connected}\}.$$

- Shortest path problem, as a decision problem: corresponds to a membership test for the language

$$\{\langle G, s, t, L\rangle \mid \text{there is an } (s, t)\text{-path of length at most } L \text{ in } G\}.$$

- Halting problem, asking if a program (Turing machine) terminates on an input w, corresponds to a membership test for the langugge

$$\{\langle M.w\rangle \mid \text{a TM } M \text{ terminates on the input } w\ \}.$$

Solving a decision problem

For a language A i.e. a decision problem, A is

- decidable if there is an algorithm (= Turing machine) which decides A.
- undecidable if there is no Turing machine which decides A.

Examples of decidable languages

- $A_{DFA} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$

Examples of decidable languages

- $A_{NFA} = \{\langle B, w\rangle \mid$ B is a NFA that accepts input string w $\}$

Examples of decidable languages

- $A_{REX} = \{\langle R, w\rangle \mid$
  R is a regular expression that generates input string w $\}$

Examples of decidable languages

- $E_{DFA} = \{\langle A \rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$

Examples of decidable languages

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

Examples of decidable languages

- $A_{CFG} = \{\langle G, w\rangle \mid \; G \text{ is a CFG that generates input string } w \; \}$

Examples of decidable languages

- $A_{PDA} = \{\langle P, w\rangle \mid$
  P is a pushdown automaton that accepts input string w $\}$; caution

Examples of decidable languages

- Any context-free language A.

Inherently looping TM

First undecidable language

Consider the language $A_{TM} = \{\langle M, w\rangle \mid \quad \text{M is a TM and } M \text{ accepts } w.\}$.

1. We know that A_{TM} is Turing-recognizable; universal Turing machine.
2. But it is undecidable.

Inherently looping TM

1. Suppose that A_{TM} is decidable, i.e. there exists TM H such that

$$H(\langle M, w\rangle) = \begin{cases} \text{ACCEPT} & \text{if } M \text{ accepts } w \\ \text{REJECT} & \text{otherwise} \end{cases}$$

2. Consider a TM D gets a description $\langle M\rangle$ of an arbitrary TM M as input, and flips the answer of H on the input $\langle M, \langle M\rangle\rangle$, i.e.

$$D(\langle M\rangle) = \begin{cases} \text{ACCEPT} & \text{if } H(\langle M, \langle M\rangle\rangle) = \text{REJECT} \\ \text{REJECT} & \text{if } H(\langle M, \langle M\rangle\rangle) = \text{ACCEPT} \end{cases}$$

3. What if we run TM D on the input $\langle D\rangle$, i.e the description of itself?

Inherently looping TM

Seen from the perspective of the diagonal argument.

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*	
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$
M_3	*reject*	*reject*	*reject*	*reject*	
M_4	*accept*	*accept*	*reject*	*reject*	
$\vdots$			$\vdots$		

Figure 4.20, Sipser 2012.

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$	$\langle D \rangle$	$\cdots$
M_1	<u>*accept*</u>	*reject*	*accept*	*reject*		*accept*	
M_2	*accept*	<u>*accept*</u>	*accept*	*accept*	$\cdots$	*accept*	$\cdots$
M_3	*reject*	*reject*	<u>*reject*</u>	*reject*		*reject*	
M_4	*accept*	*accept*	*reject*	<u>*reject*</u>		*accept*	
$\vdots$			$\vdots$		$\ddots$		
D	*reject*	*reject*	*accept*	*accept*		<u>?</u>	
$\vdots$			$\vdots$				$\ddots$

Figure 4.21, Sipser 2012.

Characterizing decidability

A language $A \subseteq \Sigma^*$ is said to be co-Turing-recognizable if its complement (i.e. $\Sigma^* \setminus A$) is Turing-recognizable.

Turing-recognizable and co-Turing-recognizable

A language is decidable if and only if it is Turing-recognizable and co-Turing-recognizable.

Characterizing decidability

A language $A \subseteq \Sigma^*$ is said to be co-Turing-recognizable if its complement (i.e. $\Sigma^* \setminus A$) is Turing-recognizable.

Turing-recognizable and co-Turing-recognizable

A language is decidable if and only if it is Turing-recognizable and co-Turing-recognizable.

- The direction ($\Rightarrow$) is straightforward.
- For the direction ($\Leftarrow$), let M_1 and M_2 be two TMs recognizing A and $\bar{A}$. Build a new TM M which runs both M_1 and M_2 simultaneously on $w \in \Sigma^*$ and outputs

$$M(w) = \begin{cases} \text{ACCEPT} & \text{if } M_1(w) = \text{ACCEPT} \\ \text{REJECT} & \text{if } M_2(w) = \text{ACCEPT} \end{cases}$$

  Clearly M decides A.

Halting problem is undecidable

Halting problem: $HALT_{TM} = \{(M, w) : M \text{ is TM and } M \text{ halts on } w\}$.

From undecidability of A_{TM}, we derive

$HALT_{TM}$ is undecidable.

Halting problem is undecidable

Halting problem: $HALT_{TM} = \{(M, w) : M \text{ is TM and } M \text{ halts on } w\}$.

From undecidability of A_{TM}, we derive

$HALT_{TM}$ is undecidable.

Proof: we crucially use that $A_{TM} = \{(M, w) : M \text{ is TM and } M \text{ accepts } w\}$ is undecidable.

Halting problem is undecidable

Suppose the contrary; let D be a decider TM for $HALT_{TM}$.

Then, we can build a decider R for A_{TM} that works as follows:

On input (M, w)

1. simulate D on (M, w).
2. if D accepts (M, w), then simulate M on w and output the answer of M on w as the answer of R.
3. if D rejects (M, w) (hence M loops on w), then reject (M, w).