

# Undecidability

Chanathip Namprempre

Faculty of Engineering  
Thammasat University

# Outline

- 1 Diagonalization
- 2 Undecidability
- 3 Mapping Reducibility
  - Definitions
  - Properties

## Theorem

*There exists a non-RE language.*

## Claim 1

The set of all languages is uncountable.

## Claim 2

The set of all TMs is countable.

## Proof

Since one TM can only recognize one language, Claim 1 and Claim 2 imply that there is a language not recognizable by a TM.

### Corollary

There is an undecidable language.

### Proof

There is an unrecognizable language  $L$ . So,  $L$  is not in RE. Since R is a subset of RE, then  $L$  is not in R.

# Countable sets

## Definition

A set is **countable** if it is finite or has the same size as  $\mathbb{Z}^+$ .

## Definition

A set  $A$  has the **same size** as  $\mathbb{Z}^+$  iff there is a one-to-one correspondence mapping elements from  $A$  to  $\mathbb{Z}^+$ .

## Theorem

*The set of even number  $E$  is countable.*

## Proof

Let  $f : \mathbb{Z}^+ \rightarrow E$  be the following function. Let  $n \in \mathbb{Z}^+$ . Then,

$$f(n) = 2n$$

Claim:  $f$  is one-to-one and onto.

So,  $E$  has the same size as  $\mathbb{Z}^+$ . So  $E$  is countable.

## Example of an uncountable set

### Theorem

*The set of reals is uncountable.*

Proof by contradiction using diagonalization.

Suppose toward a contradiction that  $\mathbb{R}$  was countable, then assume a list of all elements of  $\mathbb{R}$ . Use diagonalization to show that there is a real number that cannot be in the list, leading to a contradiction.

# Claim 1: The set of all languages is uncountable.

Show a one-to-one correspondance mapping elements from the set  $\mathcal{L}$  to  $\mathcal{B}$  where

- $\mathcal{L}$  = the set of all languages over an alphabet  $\Sigma$ .
- $\mathcal{B}$  = the set of all infinite binary sequences.

So,  $\mathcal{L}$  and  $\mathcal{B}$  are of the same size.

Prove that  $\mathcal{B}$  is uncountable using diagonalization.

So  $\mathcal{L}$  is uncountable.

## Claim 2: The set of all TMs is countable.

We know that  $\Sigma^*$  is countable. (The list is all strings of length 0, followed by all strings of length 1, etc.)

The set of all TMs is the subset of  $\Sigma^*$  where we drop the strings that are not valid encodings of TMs.

So, the set of all TMs is countable.



# Outline

- 1 Diagonalization
- 2 Undecidability**
- 3 Mapping Reducibility
  - Definitions
  - Properties

# An undecidable language

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

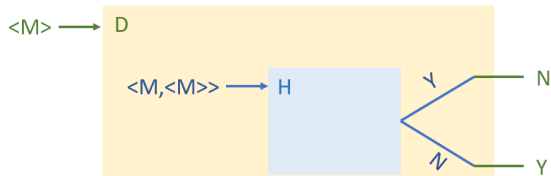
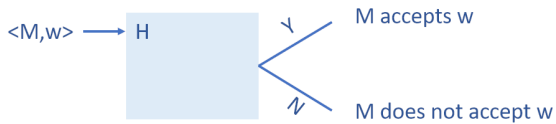
This language is recognizable (can you see why?) **BUT undecidable!**

## Proof.

We prove by contradiction. Suppose  $A_{\text{TM}}$  is decidable. Let  $H$  be the decider for  $A_{\text{TM}}$ . Let  $D$  be a TM that, on input a TM  $\langle M \rangle$ ,

- 1 runs  $H$  as a sub-routine
- 2 if  $H(\langle M, \langle M \rangle \rangle)$  rejects, then accepts
- 3 if  $H(\langle M, \langle M \rangle \rangle)$  accepts, then rejects

We get a contradiction when we run  $D(\langle D \rangle)$ . (Can you see why?) □



# Showing that $A_{TM}$ is undecidable

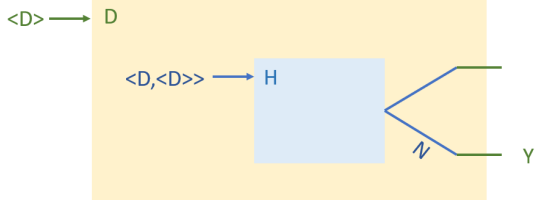
## The contradiction

Case 1: Suppose  $D(\langle D \rangle) = \text{yes}$ .

From the definition of  $D$ , this means that  $H(\langle D, \langle D \rangle \rangle) = \text{no}$ , but since  $H$  decides  $A_{TM}$ , we see that  $H$  would only say no if  $D(\langle D \rangle) = \text{no}$ . This is a contradiction.

Case 2: Suppose  $D(\langle D \rangle) = \text{no}$ .

From the definition of  $D$ , this means that  $H(\langle D, \langle D \rangle \rangle) = \text{yes}$ , but since  $H$  decides  $A_{TM}$ , we see that  $H$  would only say yes if  $D(\langle D \rangle) = \text{yes}$ . This is a contradiction.



# Diagonalization

The proof above is based on a technique called [diagonalization](#).

We can see this if we draw a table whose entries tell [whether the machine in the given row accepts the input in the given column](#) as follows.

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	...	$\langle D \rangle$	...
$M_1$	accept	reject	accept	...	accept	...
$M_2$	reject	accept	accept	...	accept	...
$M_3$	accept	accept	reject	...	accept	...
$\vdots$				$\ddots$		
$D$	reject	reject	accept	...	???	...
$\vdots$						$\ddots$

The cell [???](#) asks whether  $D$  accepts  $\langle D \rangle$ .

# Undecidability and Unrecognizability

Since  $A_{TM}$  is not recursive but is r.e., we get the following corollary.

## Corollary

$A_{TM}$  is not co-r.e.

# Languages we consider

Most of these are r.e. Some are not r.e. Some are even neither r.e. nor co-r.e. (Can you tell which are r.e.?)

## Example

$A_{TM}$	$= \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ accepts } w \}$
$Halt_{TM}$	$= \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ halts on input } w \}$
$Hang_{TM}$	$= \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ hangs on input } w \}$
$E_{TM}$	$= \{ \langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset \}$
$BlankTape_{TM}$	$= \{ \langle M \rangle \mid M \text{ is a TM and } M \text{ halts on } \varepsilon \}$
$AcceptsSome_{TM}$	$= \{ \langle M \rangle \mid M \text{ is a TM and } M \text{ accepts some input } \}$
$AcceptsAll_{TM}$	$= \{ \langle M \rangle \mid M \text{ is a TM and } M \text{ accepts all inputs } \}$
$Regular_{TM}$	$= \{ \langle M \rangle \mid M \text{ is a TM and } L(M) \text{ is a regular language} \}$
$EQ_{TM}$	$= \{ \langle M_1, M_2 \rangle \mid M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2) \}$
$All_{CFG}$	$= \{ \langle G \rangle \mid G \text{ is a CFG and } L(G) = \Sigma^* \}$
$All_{DFA}$	$= \{ \langle D \rangle \mid D \text{ is a DFA and } L(D) = \Sigma^* \}$



# Many problems are undecidable!

We do not want to go through diagonalization every time we want to show that something is undecidable.

So we relate undecidable problems to other problems.  
(This is a recurring theme in computability and complexity theory.)

We do this via **reduction**.

# Halting Problem

## Theorem

$\text{Halt}_{\text{TM}} = \{\langle M, w \rangle \mid M \text{ is a TM and } M \text{ halts on input } w\}$  is undecidable.

The **reduction** goes as follows:

Suppose toward a contradiction that  $\text{Halt}_{\text{TM}}$  is decidable. Let  $R$  be the decider for  $\text{Halt}_{\text{TM}}$ . We use  $R$  to construct a decider for  $A_{\text{TM}}$ . This is a contradiction. So  $\text{Halt}_{\text{TM}}$  is undecidable.

$S =$  “On input  $\langle M, w \rangle$ , an encoding of a TM  $M$  and a string  $w$ ,

1. Run  $R$  on input  $\langle M, w \rangle$ .
2. If  $R$  rejects, reject.
3. If  $R$  accepts, simulate  $M$  on  $w$  until it halts.
4. If  $M$  has accepted, accept. Otherwise, reject.”

Then, we show that, if  $R$  decides  $\text{Halt}_{\text{TM}}$ , then  $S$  decides  $A_{\text{TM}}$ .

One could try this approach to prove other problems undecidable.

For example, try  $E_{TM}$ .

# Outline

- 1 Diagonalization
- 2 Undecidability
- 3 Mapping Reducibility
  - Definitions
  - Properties

# Mapping reducibility: Definitions

The reduction that we saw can be formalized explicitly as a **mapping** (also known as **many-to-one**) **reduction**.

## Definition

Language  $A$  is **mapping reducible** to language  $B$ , written  $A \leq_m B$  iff there is a mapping reduction of  $A$  to  $B$ .

## Definition

A function  $f : \Sigma^* \rightarrow \Sigma^*$  is a **mapping reduction** of a language  $A$  to a language  $B$  iff  $f$  is computable and, for every  $w \in \Sigma^*$ ,

- If  $w \in A$ , then  $f(w) \in B$ , and
- If  $w \notin A$ , then  $f(w) \notin B$ .

Try drawing a picture to visualize this property of  $f$ .

## Mapping reducibility: Definitions (cont.)

### Definition

A function  $f : \Sigma^* \rightarrow \Sigma^*$  is a **computable function** iff there is a TM  $M$  that computes it.

### Definition

A TM  $M$  **computes** a function  $f$  iff, for any input  $w \in \Sigma^*$ , the machine  $M$  halts with just  $f(w)$  on its tape.

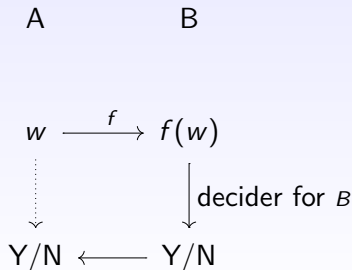
# Why do we care about mapping reductions?

A reduction helps us relate different problems to each other.

For example, suppose we know that  $A \leq_m B$  and that  $B$  is decidable.

Then, we know that  $A$  is also decidable. WHY??

We can decide  $A$  like so.



# Properties we get from mapping reducibility

Let  $A, B$  be languages. Suppose  $A \leq_m B$ .

- ➊ If  $B$  is recursive, then so is  $A$ .
- ➋ If  $A$  is not recursive, then neither is  $B$ .
- ➌ If  $B$  is recursively enumerable, then so is  $A$ .
- ➍ If  $A$  is not recursively enumerable, then neither is  $B$ .

Also, these statements are true:

- $A \leq_m B$  iff  $\overline{A} \leq_m \overline{B}$
- If  $A \leq_m B$  and  $B \leq_m C$ , then  $A \leq_m C$ .  
 (i.e.  $\leq_m$  is a transitive relation.)

Questions: Is  $\leq_m$  reflexive? Is it symmetric?



# Applying mapping reducibility

The properties of mapping reducibility are useful.

For example, if you want to show that a language

- $L$  is not r.e., then you can show that  $A_{\text{TM}} \leq_m \bar{L}$
- $L$  is not co-r.e., then you can show that  $A_{\text{TM}} \leq_m L$
- $L$  is undecidable, then you can show either  
 (but choose carefully)

# Try writing reductions for these

## Example

- $A_{TM} \leq_m \text{Halt}_{TM}$
- $A_{TM} \leq_m \overline{E_{TM}}$
- $E_{TM} \leq_m EQ_{TM}$
- $EQ_{TM}$  is neither r.e. nor co-r.e.  
(Try proving that  $A_{TM} \leq_m \overline{EQ_{TM}}$  and  $A_{TM} \leq_m EQ_{TM}$ )
- $\text{Halt}_{TM} \leq_m \text{BlankTape}_{TM}$
- $\text{Halt}_{TM} \leq_m \text{AcceptsSome}_{TM}$
- $\text{Halt}_{TM} \leq_m \text{AcceptsAll}_{TM}$
- $\text{BlankTape}_{TM} \leq_m \text{Halt}_{TM}$
- $\text{AcceptsSome}_{TM} \leq_m \text{Halt}_{TM}$
- $\text{AcceptsAll}_{TM} \leq_m \text{Halt}_{TM}$  (You will fail.)
- $\text{AcceptsAll}_{TM} \leq_m EQ_{TM}$

What conclusions can you draw from each of these reductions?