

# Diagonalization and decidability problems

## Textbook: Chapter 4

$s_1$	=	0	0	0	0	0	0	0	0	0	0	0	0	...
$s_2$	=	1	1	1	1	1	1	1	1	1	1	1	1	...
$s_3$	=	0	1	0	1	0	1	0	1	0	1	0	...	
$s_4$	=	1	0	1	0	1	0	1	0	1	0	1	...	
$s_5$	=	1	1	0	1	0	1	1	0	1	0	1	...	
$s_6$	=	0	0	1	1	0	1	1	0	1	1	0	...	
$s_7$	=	1	0	0	0	1	0	0	1	0	0	...		
$s_8$	=	0	0	1	1	0	0	1	1	0	0	1	...	
$s_9$	=	1	1	0	0	1	1	0	0	1	1	0	...	
$s_{10}$	=	1	1	0	1	1	1	0	0	1	0	1	...	
$s_{11}$	=	1	1	0	1	0	1	0	0	1	0	...		
$\vdots$		$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	

$s$	=	1	0	1	1	1	0	1	0	0	1	1	...
-----	---	---	---	---	---	---	---	---	---	---	---	---	-----

## Sizing infinite sets

- ▶ The size of a finite set is the number of elements in it
- ▶ What about infinite sets?
- ▶ We can show that a set  $A$  is the same size as another,  $B$ , by making a **1-to-1 mapping** from each element of  $A$  to  $B$
- ▶ This works even for infinite sets!

## Examples

**Ex:** An infinite amount of \$100 bills and an infinite amount of \$1 bills **are worth the same amount.**

**Ex:** Are there more integers ( $\mathbb{Z}$ ) or even integers ( $2\mathbb{Z}$ )? It would seem that, since only half of all integers are even, that there would be more of them. However, **this isn't true!**

Let  $f : \mathbb{Z} \rightarrow 2\mathbb{Z}$  be the function  $f(k) = 2k$ . This is a **bidirectional mapping**, so the sets it maps to and from must be the same size!

## Different infinity sizes

**Ex:** Is the set of all natural numbers  $\{0, 1, 2, 3, \dots\}$  the same size as the set of all real numbers (EG  $\pi, 4, \frac{1}{2}, e, \dots$ )? They are both infinite, so one would assume so.

As it turns out, **no!** Some infinities are larger than others.

## Cantor's diagonalization proof for $\mathbb{N}$ and $\mathbb{R}$

**Thm:** There are more real numbers between 0 and 1 than natural numbers 0 to  $\infty$ .

**Pf:** We will try to construct an arbitrary-order table where the first column is an increasing integer index starting at 0 and the second column is a list of all the real numbers from 0 to 1. If there are the same number, all reals will be contained in this list.

Index	Real number
0	$c_0 = 0.12345678 \dots$
1	$c_1 = 0.31415926 \dots$
2	$c_2 = 0.27182818 \dots$
3	$c_3 = 0.00000000 \dots$
4	$c_4 = 0.99999999 \dots$
$\vdots$	$\vdots$

## Cantor pt. 2

Now we construct a new number  $c'$ . The  $i$ -th decimal digit of  $c'$  will be one more than the  $i$ -th digit of  $c_i$  (or 0 if it was 9). By definition,  $c'$  is not in our list (it is different from every element in at least one digit)!

$$c_0 = 0.\mathbf{1}2345\dots$$

$$c_1 = 0.3\mathbf{1}415\dots$$

$$c_2 = 0.27\mathbf{1}82\dots$$

$$c_3 = 0.0000\mathbf{0}\dots$$

$$c_4 = 0.9999\mathbf{9}\dots$$

$$\vdots$$

$$c' = 0.\mathbf{22210}\dots$$

Since we already used every natural number for indices, **there must be more real numbers than natural ones**. End of proof.

**Note:** We used the range  $[0, 1]$ , but this holds for any  $[a, b] : a, b \in \mathbb{R}$ ! There are more real numbers between any two real numbers than there are natural numbers.

# Infinity sizes

**Def:** An infinite set is said to be **countable** iff there exists a correspondence between it and the natural numbers.

**Def:** An infinite set is said to be **uncountable** iff there exists a correspondence between it and the real numbers.

**Corollary:** Any uncountable set is larger than any countable set.  
Any countable set is larger than any finite set.

# Proving unrecognizable languages exist

**Thm:** There are more languages on any  $\Sigma$  than there are Turing machines.

**Pf:** We will prove the set of all Turing machines is countably infinite, while the set of all languages on  $\Sigma$  is uncountably infinite. Therefore, there are undecidable languages on every alphabet.



## Encoding TMs in finite binary strings

**Def:** An **encoding** onto alphabet  $\Sigma$  of a TM  $M$  is some finite string  $\langle M \rangle \in \Sigma^*$  which can be used by a UTM to simulate  $M$ .

**This is intentionally vague!** Just convince yourself that such a string exists and is finite for all valid TMs.

**Def:** A binary encoding of a TM encodes it onto alphabet  $\{0,1\}$ .

## Language characteristic strings

- ▶ A language on an alphabet  $\Sigma$  is some (possibly infinite) subset of  $\Sigma^*$
- ▶ Without loss of generality, let us assume  $\Sigma = \{0, 1\}$  (the smallest useful alphabet)
- ▶ Then  $\Sigma^*$  is  $\{\epsilon, 0, 1, 01, 10, 11, \dots\}$
- ▶ Let  $\sigma_i$  be the  $i$ -th element of  $\Sigma^*$  (0-indexed)
- ▶ **Note:** Every  $\sigma_i$  is of finite length!

**Def:** The characteristic string  $c_L$  of language  $L$  on alphabet  $\Sigma$  is an **infinite** binary string  $c = c_0c_1c_2 \cdots c_i \cdots$  where  $c_i = 1$  iff  $\sigma_i \in L$ .

- ▶ Ex:  $c_{\emptyset} = 0000000 \dots$
- ▶ Ex:  $c_{\Sigma^*} = 1111111 \dots$
- ▶ Ex:  $c_{\{\epsilon\}} = 1000000 \dots$
- ▶ Ex:  $c_{\{1, 11\}} = 0010001 \dots$

## Diagonalizing the TMs

Create a table such that:

- ▶ Every TM  $M_i$ 's binary encoding  $a_i$  is in one column
- ▶ The characteristic string  $c_i$  of  $\mathcal{L}(M_i)$  (the set all strings *recognized* by  $M_i$ ) is in the other
- ▶ By convention, let any invalid TM encoding have a characteristic string of all 0 (follows from UTM def.)

TM Binary encoding	Characteristic string
$a_0 = 00000$	$c_0 = 01010101\dots$
$a_1 = 111111111$	$c_1 = 11111111\dots$
$a_2 = 0$	$c_2 = 00000000\dots$
$a_3 = 1111\dots1111$	$c_3 = 10101111\dots$
$\vdots$	$\vdots$

**Every TM is listed in the left column.**

## Diagonalization pt. 2

Let  $c'$  be an infinite binary characteristic string. Let the  $i$ -th bit of  $c'$  be the negation of the  $i$ -th bit of  $c_i$ .

TM Binary encoding	Characteristic string
$a_0 = 0000 \dots 0000$	$c_0 = \mathbf{0}101 \dots$
$a_1 = 11111111$	$c_1 = \mathbf{1}111 \dots$
$a_2 = 0$	$c_2 = 00\mathbf{00} \dots$
$a_3 = 1111 \dots 1111$	$c_3 = 101\mathbf{0} \dots$
$\vdots$	$\vdots$

$$c' = \mathbf{1011} \dots$$

**The language of  $c'$  is not recognized by the list!** Since the list has every TM, no TM recognizes  $c'$ : It is said to be **unrecognizable**.

**Therefore, some languages are not recognizable.**

## Proving the acceptance problem is recognizable

**Def:** The acceptance problem. Given some TM  $M$  and input  $w$ , does  $M$  accept? We define the language  $A_{TM}$  ( $A$  for “accept”) to be:

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

**Thm:** The acceptance problem ( $A_{TM}$ ) is *recognizable* (a positive answer can be given, but not always in finite time).

**Pf:** By construction. Let  $A$  be a UTM taking input  $\langle M, w \rangle$  and accepting only if  $M(w)$  halts. Being a UTM, it can simulate  $M(w)$ . If  $M(w)$  ever halts,  $A$  accepts. By definition,  $A$  recognizes (but doesn't decide)  $A_{TM}$ . End of proof.

# Proving the acceptance problem is undecidable

**Thm:**  $A_{TM}$  is undecidable.

**Pf:** By contradiction. Assume  $A_{TM}$  is decidable. Then there exists a TM  $H$  **deciding**  $A_{TM}$ . We will derive a contradiction similar to Russell's paradox.

$$H(\langle M, w \rangle) = \begin{cases} \text{accept} & \text{if } M \text{ accepts } w \\ \text{reject} & \text{if } M \text{ rejects or loops on } w \end{cases}$$

Let  $D$  be a new TM using  $H$  as a subroutine (this is legal). Since  $H$  is a decider, it always completes in finite time. We will design  $D$  to be a decider as well.

## Acceptance problem undecidability pt. 2

$D =$  “On input  $\langle M \rangle$ , where  $M$  is a TM:

1. Run  $H$  on input  $\langle M, \langle M \rangle \rangle$  (determine whether  $M$  halts when given its own description)
2. If  $H$  accepted, **reject**. If  $H$  rejected, **accept**.”

This means that  $D(\langle M \rangle)$  accepts if  $M(\langle M \rangle)$  rejects or loops and rejects if  $M(\langle M \rangle)$  accepts.

**What happens if we run  $D(\langle D \rangle)$ ?**

## Acceptance problem undecidability pt. 3

$$D(\langle D \rangle) = \begin{cases} \text{accept} & \text{if } D \text{ rejects or loops on } \langle D \rangle \\ \text{reject} & \text{if } D \text{ accepts } \langle D \rangle \end{cases}$$

- ▶ If  $D(\langle D \rangle)$  accepts:
  - ▶ By definition, this means it must reject
  - ▶ **Contradiction!**
- ▶ If  $D(\langle D \rangle)$  rejects:
  - ▶ By definition, this means it must accept
  - ▶ **Contradiction!**

Therefore,  $D(\langle D \rangle)$  accepts if and only if it does not: A contradiction.  
The assumption that  $A_{TM}$  is decidable must be false. End of proof.

**Proven by Church and Turing in 1936.**



# Co-Turing-recognizability

**Def:** A language  $L$  is said to be **co-Turing-recognizable** if  $\bar{L}$  is Turing-recognizable.

**Thm:** A language is decidable iff it is Turing-recognizable and co-Turing-recognizable. (pf excluded)

**Corollary:** A language is undecidable iff it or its complement are not Turing-recognizable.

**Note:** All languages which are Turing *decidable* are trivially co-Turing-decidable and vice versa.

## A Turing-unrecognizable language

**Thm:**  $\overline{A_{TM}}$  is not Turing-recognizable.

**Pf:** Since  $A_{TM}$  is undecidable, either it or its complement must be Turing-unrecognizable. We already proved  $A_{TM}$  is Turing-recognizable. Therefore,  $\overline{A_{TM}}$  must be Turing-unrecognizable. End of proof.

Next time: Reducibility