

# UNDECIDABILITY

THURSDAY SEP 29

**Definition:** A Turing Machine is a 7-tuple  $T = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ , where:

$Q$  is a finite set of states

$\Sigma$  is the input alphabet, where  $\square \notin \Sigma$

$\Gamma$  is the tape alphabet, where  $\square \in \Gamma$  and  $\Sigma \subseteq \Gamma$

$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$

$q_0 \in Q$  is the start state

$q_{\text{accept}} \in Q$  is the accept state

$q_{\text{reject}} \in Q$  is the reject state, and  $q_{\text{reject}} \neq q_{\text{accept}}$

A TM recognizes a language if it accepts all and only those strings in the language

A language is called Turing-recognizable or recursively enumerable if some TM recognizes it

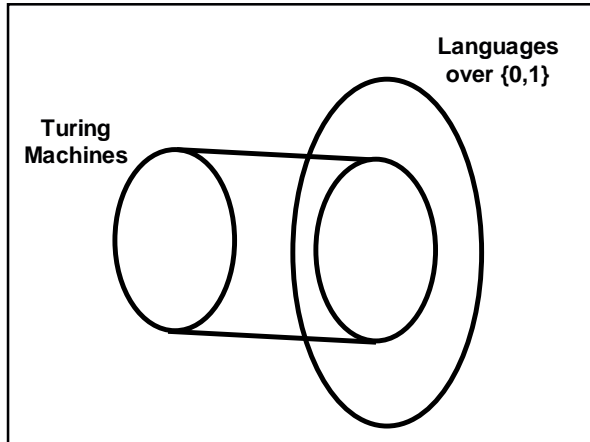
A TM decides a language if it accepts all strings in the language and rejects all strings not in the language

A language is called decidable or recursive if some TM decides it

## There are languages over $\{0,1\}$ that are not decidable

If we believe the Church-Turing Thesis, this is MAJOR: It means there are things that computers inherently cannot do

We will prove this using a simple counting argument. We will show there is no onto function from the set of all Turing Machines to the set of all languages over  $\{0,1\}$ .



**Theorem:** There is no onto function from the positive integers to the real numbers between 0 and 1 (exclusive)

**Proof:** Suppose  $f$  is such a function:

1	→	0.28347279...
2	→	0.88388384...
3	→	0.77635284...
4	→	0.11111111...
5	→	0.12345678...
:		:

[ n-th digit of  $r$  ] =  $\begin{cases} 1 & \text{if [ n-th digit of } f(n) \text{ ] } \neq 1 \\ 0 & \text{otherwise} \end{cases}$

$f(n) \neq r$  for all  $n$

Let  $L$  be a set and  $2^L$  be the power set of  $L$

**Theorem:** There is no onto map from  $L$  to  $2^L$

**Proof:** Assume, for a contradiction, that there is an onto map  $f : L \rightarrow 2^L$

Let  $S = \{ x \in L \mid x \notin f(x) \}$

If  $S = f(y)$  then  $y \in S$  if and only if  $y \notin S$

**No matter what,  $2^L$  always has more elements than  $L$**

Let  $Z^+ = \{1,2,3,4,\dots\}$ . There exists a bijection between  $Z^+$  and  $Z^+ \times Z^+$

(1,1) (1,2) (1,3) (1,4) (1,5) ...  
(2,1) (2,2) (2,3) (2,4) (2,5) ...  
(3,1) (3,2) (3,3) (3,4) (3,5) ...  
(4,1) (4,2) (4,3) (4,4) (4,5) ...  
(5,1) (5,2) (5,3) (5,4) (5,5) ...

No matter what,  $2^L$  always has more elements than  $L$

Not all languages over  $\{0,1\}$  are decidable

Turing Machines	Languages over $\{0,1\}$
Strings of 0s and 1s	Sets of strings of 0s and 1s

### THE ACCEPTANCE PROBLEM

$A_{TM} = \{ (M,w) \mid M \text{ is a TM that accepts string } w \}$

Theorem:  $A_{TM}$  is semi-decidable (r.e.) but NOT decidable

$A_{TM}$  is semi-decidable:

Define TM  $U$  as follows:

On input  $(M,w)$ ,  $U$  runs  $M$  on  $w$ . If  $M$  ever accepts, accept. If  $M$  ever rejects, reject.

$A_{TM} = \{ (M,w) \mid M \text{ is a TM that accepts string } w \}$

$A_{TM}$  is undecidable: (proof by contradiction)

Assume machine H decides  $A_{TM}$

$$H( (M,w) ) = \begin{cases} \text{Accept} & \text{if } M \text{ accepts } w \\ \text{Reject} & \text{if } M \text{ does not accept } w \end{cases}$$

Construct a new TM D as follows: on input M, run H on (M,M) and output the opposite of H

$$D( D ) = \begin{cases} \text{Reject} & \text{if } D \text{ accepts } D \\ \text{Accept} & \text{otherwise} \end{cases}$$

### OUTPUT OF H

	$M_1$	$M_2$	$M_3$	$M_4 \dots$	D
$M_1$	accept	accept	accept	reject	accept
$M_2$	reject	accept	reject	reject	reject
$M_3$	accept	reject	reject	accept	accept
$M_4$	accept	reject	reject	reject	accept
:					
D	reject	reject	accept	accept	?

Theorem:  $A_{TM}$  is semi-decidable (r.e.) but NOT decidable

Theorem:  $\neg A_{TM}$  is not even semi-decidable!

**WWW.FLAC.WS**

Read chapter 4 of the book for next time