

Can we solve  
EVERY "computational"  
problem by computers?

specify:

- what problems?
- what computers?

otherwise,  
question too vague

What problems?

Consider YES/NO  
questions about  
finite strings

- 0/1 strings
- strings using  
symbols from a  
finite set  $\Sigma$

example:  $\Sigma = \{a, b, c\}$

strings:  
(words)       $abcbbac$

YES/NO questions

about membership:

$$L \subseteq \Sigma^*$$

< set of strings

where answer is YES

given  $w \in \Sigma^*$

is  $w$  in  $L$ ?

(YES or NO)



We can view  
all DECISION  
PROBLEMS we studied  
this way:

- 3-SAT:  
given 3-CNF formula  
(string of symbols  
from  $\Sigma = \{\wedge, \vee, x_i, \bar{x}_i, (x_1 \dots x_n \bar{x}_1 \dots \bar{x}_n)\}$ )  
YES/NO: does it have  
satisfying assignment?

## • CLIQUE

given a graph  $G$   
and integer  $k$   
can represent  
graphs by 0/1 strings!

YES/NO :

does it have a  
clique of size  $\geq k$ ?

What computers?

EVERY computer??

modify question:

Can we solve every  
computational problem  
(e.g. YES/NO questions)

by an algorithm?

What algorithms?

What is an algorithm?



# Turing:

formalize notion  
of algorithms

## Turing machine:

$\Sigma$ : finite set of symbols

tape: infinite

head: looks at a  
symbol on tape  $\rightarrow$

- change symbol
- move left/right/stay
- change STATE

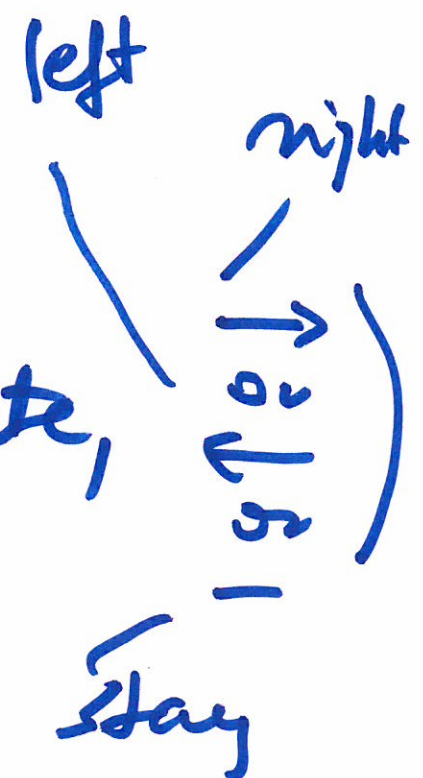
# Specify Turing Machine:

$\Sigma$  : finite set of symbols

$Q$  : finite set of states

Transition Table:

given a pair  
(symbol, state)

$\rightarrow$  (symbol, state, )



# Description of a TM:

string of symbols  
from a finite set

Ask questions (YES/NO)  
about strings  
that are descriptions  
of Turing Machines

# HALTING PROBLEM

Input :  $M, x$  — string  
description of  
Turing Machine

YES/NO question:

answer YES,  
if  $M$  HALTS on  $x$

Cannot be solved  
by ANY "algorithm"

Suppose  $\hat{M}$  solves  
halting problem:

$$\hat{M}(M, x) = \begin{cases} \text{YES} & \text{if } M \text{ halts on } x \\ \text{NO} & \text{otherwise} \end{cases}$$

Consider  $\tilde{M}$   
(if  $\hat{M}$  exists,  $\tilde{M}$  also exists)

$\tilde{M}$  on input  $M$

- first, simulates  $\hat{M}$  on  $M, M$

- next,

if  $\hat{M}(M, M) = \text{NO}$ ,  $\tilde{M}(M)$  STOPS

if  $\hat{M}(M, M) = \text{YES}$ ,  $\tilde{M}(M)$   
keeps running forever



But this gives a  
contradiction.

Such machine  $\tilde{M}$  cannot  
exist.

$\Rightarrow \hat{M}$  cannot exist either.

To see that  $\tilde{M}$  cannot  
exist, consider  
what is  $\tilde{M}$  supposed to do  
on its own description.

What would  $\tilde{M}$  do on input  $\tilde{M}$ ?



If  $\tilde{M}$  stops on  $\tilde{M}$ ,  
then  $\hat{M}(\tilde{M}, \tilde{M}) = \text{YES}$ ,  
and  $\tilde{M}$  on  $\tilde{M}$  would  
keep running forever

— contradiction  
(so  $\tilde{M}$  cannot stop on  $\tilde{M}$ )

If  $\tilde{M}$  does not stop on  $\tilde{M}$   
then  $\hat{M}(\tilde{M}, \tilde{M}) = \text{NO}$ ,  
and  $\tilde{M}$  on  $\tilde{M}$  should STOP.

— contradiction again.



$\tilde{M}$  cannot exist.