LECTURE - 9

"TURING MACHINES"

Regular Languages
DFA / NFAs

- string
- concat
- choice
- looping

$0^n 1^n$ is not regular,
but is context free.

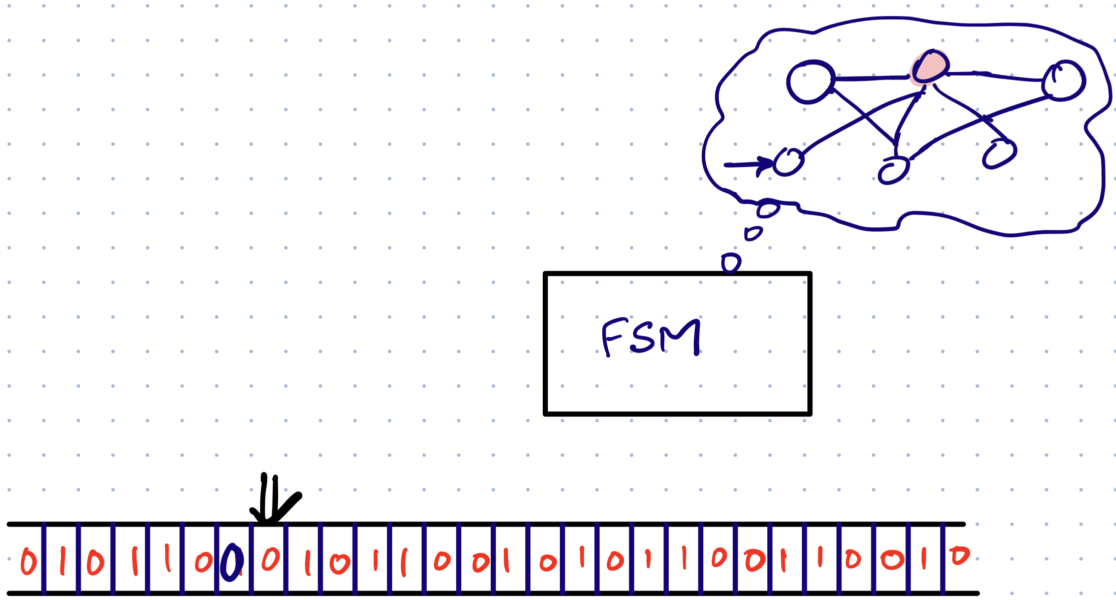
Context free languages

- recursion

$0^n 1^n 2^n$

Turing Machine

FSM with access to unrestricted memory



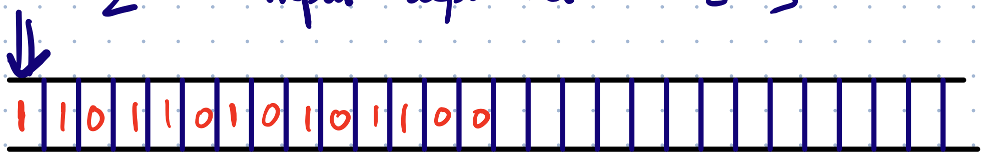
TURING MACHINE

Q - finite set of states

Start - start state

accept
reject } halt states

Σ - input alphabet $\{0,1\}$



Γ - tape alphabet $\Sigma \subseteq \Gamma$

$\square$ - blank symbol $\in \Gamma$.

$$\delta : (Q \setminus \{\text{accept}, \text{reject}\}) \times \Gamma \longrightarrow Q \times \Gamma \times \{-1, +1\}$$

Configuration $(q, x, i) \in Q \times \Gamma^* \times \mathbb{N}$

current state

string on tape

position of head

$$(p, x, i) \Rightarrow (q, y, j)$$

$$\delta(p, \underset{\text{TAPE}}{a}) = (q, b, +1)$$

$$(p, xay, i) \Rightarrow (q, xby, i+1)$$

RAM : Random Access Memory.

Any language decidable in time $T(n)$ by a RAM is decidable in time $(T(n))^4$ by TM.

VARIATIONS

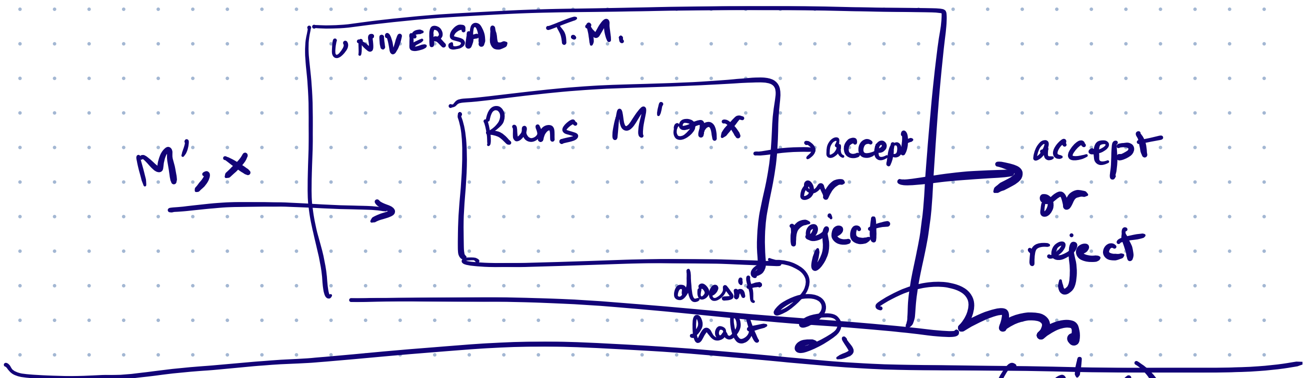
- many accepting / rejecting states.
- either write or move head.
- doubly infinite tape
- many heads
- insert and delete cells
- multiple tapes each with their own head.
- random access.

Hilbert :

Want an algorithm, s.t. if I feed a theorem, it will tell me whether theorem is true or false.

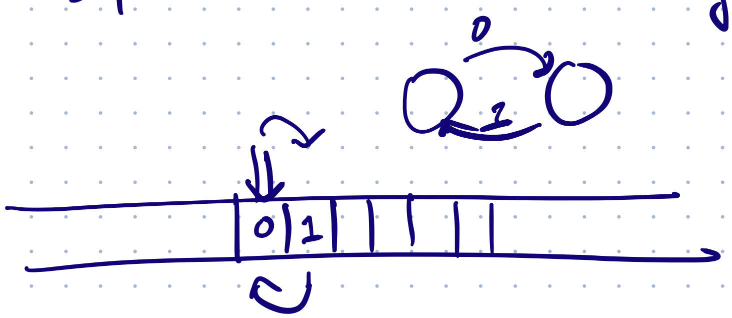
ALAN TURING : such an algorithm cannot exist.
(Also Gödel's incompleteness theorems).

TURING MACHINES



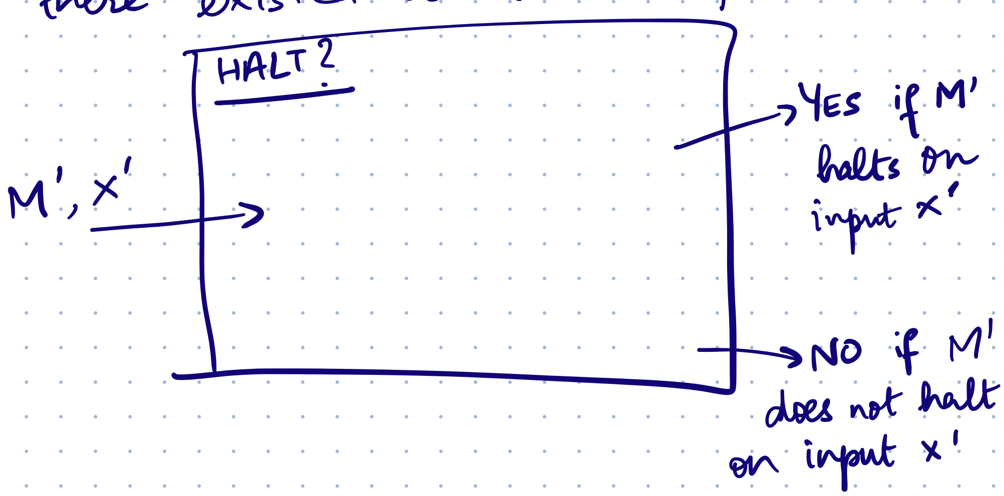
Input to universal T.M. is (M', x)

Input to M' is the string x.

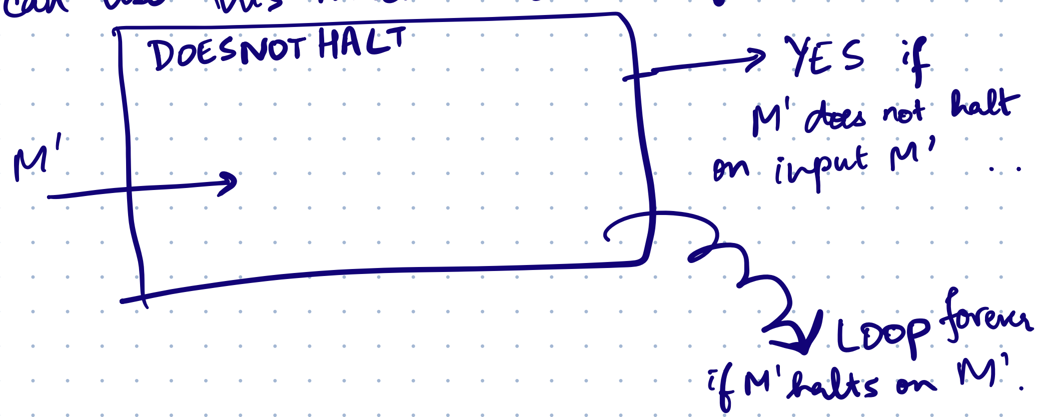


Can feed a turing machine M its own description as input.

Suppose there existed a machine for HALT.



Then I can use this machine to build:



What does DOESNOT HALT do on input

M' = DOESNOTHALT ?

- Suppose it outputs YES.

$\Rightarrow$ This means DOESNOT HALT halted and output YES on input DOESNOTHALT.

But it should only do this if M' = DOESNOTHALT does not halt on input M'. $\Rightarrow\!\times\!\Leftarrow$ CONTRADICTION.

- Suppose it loops forever.

$\Rightarrow$ This means DOESNOTHALT doesn't halt on input DOESNOTHALT.

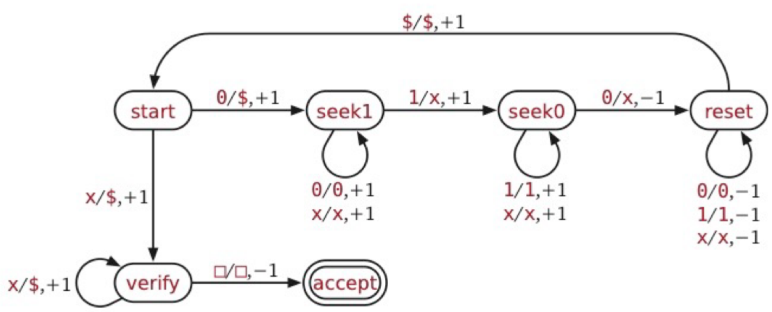
$\Rightarrow$ M' on input M' (= DOESNOTHALT) does not halt.

$\Rightarrow$ It should've halted and output YES. $\Rightarrow\!\times\!\Leftarrow$ CONTRADICTION

TURING MACHINES CANNOT DECIDE WHETHER A T.M. HALTS OR NOT.

$\delta(\;p\;, a) = (\;q\;, b, \delta\;)$	explanation
$\delta(\text{start}, \texttt{0}) = (\text{seek1}, \texttt{\$}, +1)$	mark first 0 and scan right
$\delta(\text{start}, \texttt{x}) = (\text{verify}, \texttt{\$}, +1)$	looks like we're done, but let's make sure
$\delta(\text{seek1}, \texttt{0}) = (\text{seek1}, \texttt{0}, +1)$	scan rightward for 1
$\delta(\text{seek1}, \texttt{x}) = (\text{seek1}, \texttt{x}, +1)$	
$\delta(\text{seek1}, \texttt{1}) = (\text{seek0}, \texttt{x}, +1)$	mark 1 and continue right
$\delta(\text{seek0}, \texttt{1}) = (\text{seek0}, \texttt{1}, +1)$	scan rightward for 0
$\delta(\text{seek0}, \texttt{x}) = (\text{seek0}, \texttt{x}, +1)$	
$\delta(\text{seek0}, \texttt{0}) = (\text{reset}, \texttt{x}, +1)$	mark 0 and scan left
$\delta(\text{reset}, \texttt{0}) = (\text{reset}, \texttt{0}, -1)$	scan leftward for \$
$\delta(\text{reset}, \texttt{1}) = (\text{reset}, \texttt{1}, -1)$	
$\delta(\text{reset}, \texttt{x}) = (\text{reset}, \texttt{x}, -1)$	
$\delta(\text{reset}, \texttt{\$}) = (\text{start}, \texttt{\$}, +1)$	step right and start over
$\delta(\text{verify}, \texttt{x}) = (\text{verify}, \texttt{\$}, +1)$	scan right for any unmarked symbol
$\delta(\text{verify}, \square) = (\text{accept}, \square, -1)$	success!

The transition function for a Turing machine that decides the language $\{\texttt{0}^n\texttt{1}^n\texttt{0}^n \mid n \geq 0\}$.



p	a	q	b	δ
start	0	seek1	$	+1
start	x	verify	$	+1
seek1	0	seek1	0	+1
seek1	x	seek1	x	+1
seek1	1	seek0	x	+1
seek0	1	seek0	1	+1
seek0	x	seek0	x	+1
seek0	0	reset	x	+1
reset	0	reset	0	−1
reset	1	reset	1	−1
reset	x	reset	x	−1
reset	$	start	$	+1
verify	x	verify	$	+1
verify	□	accept	□	−1

(start, 001100)
⇒ (seek1, $01100)
⇒ (seek1, $01100)
⇒ (seek0, $0x100)
⇒ (seek0, $0x100)
⇒ (reset, $0x1x0)
⇒ (reset, $0x1x0)
⇒ (reset, $0x1x0)
⇒ (reset, $0x1x0)
⇒ (start, $0x1x0)
⇒ (seek1, $$x1x0)
⇒ (seek1, $$x1x0)
⇒ (seek0, $$xxx0)
⇒ (seek0, $$xxx0)
⇒ (reset, $$xxxx)
⇒ (reset, $$xxxx)
⇒ (reset, $$xxxx)
⇒ (reset, $$xxxx)
⇒ (start, $$xxxx)
⇒ (verify, $$$xxx)
⇒ (verify, $$$$xx)
⇒ (verify, $$$$$x)
⇒ (verify, $$$$$$□)
⇒ (accept, $$$$$$) ⇒ **accept!**

(start, 00100)
⇒ (seek1, $0100)
⇒ (seek1, $0100)
⇒ (seek0, $0x00)
⇒ (reset, $0xx0)
⇒ (reset, $0xx0)
⇒ (reset, $0xx0)
⇒ (start, $0xx0)
⇒ (seek1, $$xx0)
⇒ (seek1, $$xx0)
⇒ (seek1, $$xx0) ⇒ **reject!**