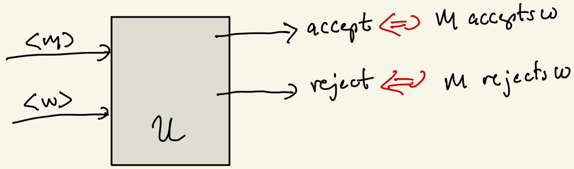
Sipser 4.2 Undecidability

$A_{TM} = \{ \langle M \rangle \langle w \rangle \mid M \text{ is a deterministic TM and } w \in L(M) \}$

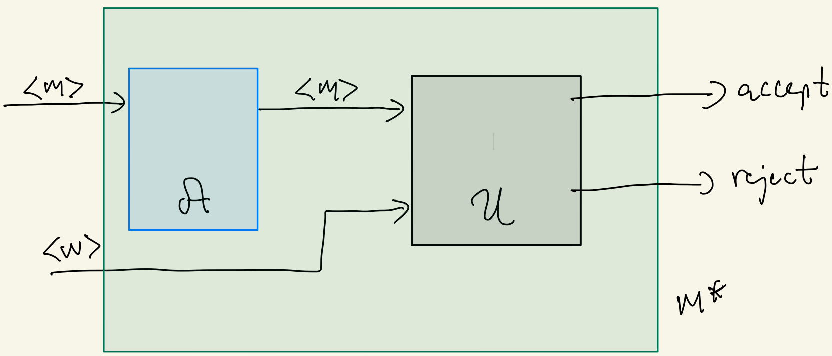
Theorem 4.11 A_{TM} is Turing-recognizable

proof: 1. check whether $\langle M \rangle$ codes a DTM
if not reject $\langle M \rangle \langle w \rangle$

2.



$\mathcal{U}$ is the universal TM
it will loop on $\langle M \rangle \langle w \rangle \Longleftrightarrow$ M loops on w



$\mathcal{A}$: check whether M is a DTM
if not loop
else send $\langle M \rangle$ to $\mathcal{U}$

M^* recognizes A_{TM}.

How can we prove that A_{TM} is not decidable?

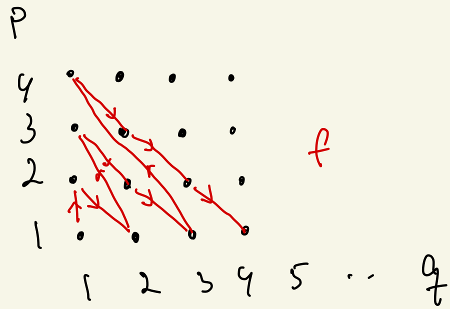
Definition 4.14 A set S is countable if either

1. S is finite, or
2. $\exists\, f : S \to \mathbb{N}$ s.t. f is 1-1 and onto

Example

- $S = \{2k \mid k \in \mathbb{N}\}$ $\quad f : 2k \to k$
- $S = \Sigma^*$ $\quad f$ orders strings lexicographically
- $S = \mathbb{Q}$ rational numbers

$$s = \frac{p}{q}$$



The set S of all binary strings is countable

a) $S = \{0,1\}^*$

$\varepsilon, 0, 1, 01, 11, 001 \dots$

Theorem The set B of infinite binary strings is uncountable

Proof: suppose $b_1, b_2, b_3 \dots b_k, b_{k+1} \dots$ is a list of all infinite binary strings.

Define the infinite binary string b^* as follows, where $b^*(i)$ is the i'th place in b^*

$$b^*(i) = \begin{cases} 1 & \text{if } b_i(i) = 0 \\ 0 & \text{if } b_i(i) = 1 \end{cases}$$

Now b^* cannot be in the list above:

suppose $b^* = b_j$, then $b^*(j) = 1 - b_j(j) \neq b_j(j)$ $\lightning$

$\square$

Observation: Every language over an alphabet Σ is a subset of $\mathcal{P}(\Sigma^*)$ = set of all subsets of Σ^* and with respect to the lexicographic ordering

$w_1, w_2, w_3, \ldots$

of Σ^* each language L over Σ corresponds 1-1 to a unique infinite binary string b_L

when

$$b_L(i) = \begin{cases} 1 & \text{if and only if } w_i \in L \\ 0 & \text{if } \quad -\,\text{"}\,- \quad w_i \notin L \end{cases}$$

Corollary The set of all languages over a non-trivial alphabet Σ is uncountable. ($|\Sigma| \geq 1$)

Recall that Turing machines can be coded over the universal alphabet and state set plus a few extra symbols

Let $A = \{a_1, a_2, \ldots\}$, $Q = \{q_1, q_2, \ldots\}$ and

$$X = \{ '(', ')', 'q', 'a', ',', '1', '0', 'R', 'L', 'S' \}$$

Then with $\Sigma = X$ we can code all TM

Consider the lexicographic ordering of strings in Σ

$$w_1, w_2, w_3 \;\ldots\; \underset{\langle M_1 \rangle}{w_{p_1}}, \;\ldots\; \underset{\langle M_2 \rangle}{w_{p_2}} \;\ldots\; \underset{\langle M_3 \rangle}{w_{p_3}} \;\ldots$$

This induces an ordering of all (codes of) Turing machines

$\langle M_1 \rangle, \langle M_2 \rangle, \ldots$

Conclusion: There are countably many Turing machines

- Each Turing machine $\langle M_i \rangle$ recognizes exactly one language, namely $L(M_i)$

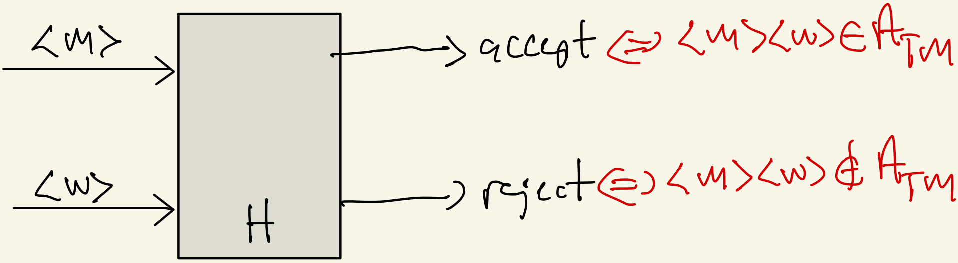
- Recall that the number of languages over an alphabet with at least 2 symbols is Uncountable.

- Hence there are (∞) many languages that are NOT Turing-recognizable.

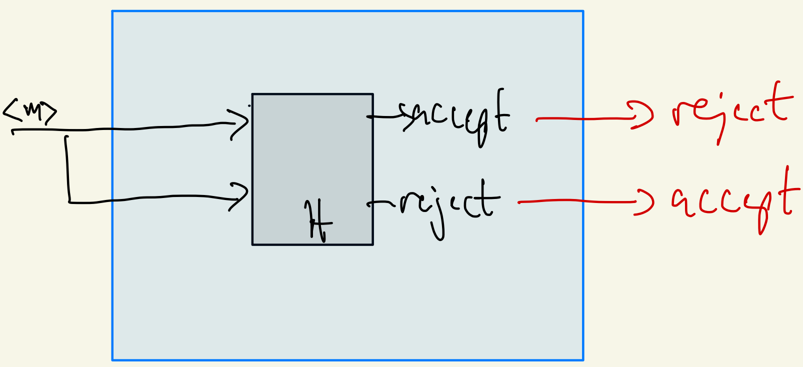
Theorem 4.11 A_{TM} is undecidable

$$A_{TM} = \{\langle M\rangle\langle w\rangle \mid M \text{ is a TM and } w \in L(M)\}$$

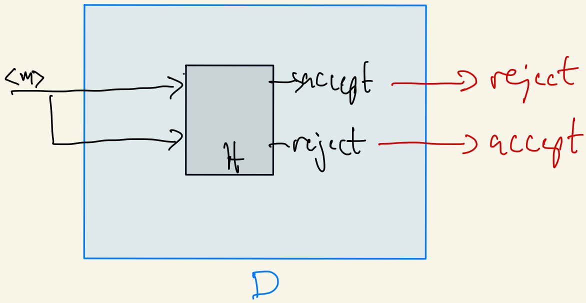
P: Suppose H is a TM which decides A_{TM}



Then we can use H to build the TM D



D



$D(\langle M \rangle)$: D's action on input $\langle M \rangle$

$$D(\langle M \rangle) = \begin{cases} \text{accept if } \langle M \rangle \notin L(M) \\ \text{reject if } \langle M \rangle \in L(M) \end{cases}$$

Hmm

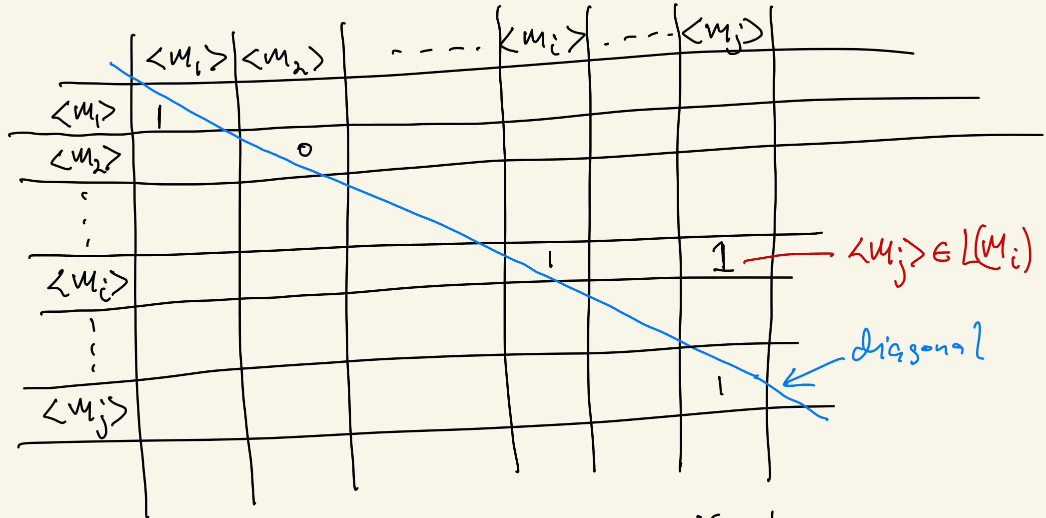
$$D(\langle D \rangle) = \begin{cases} \text{accept if } \langle D \rangle \notin L(D) \\ \text{reject if } \langle D \rangle \in L(D) \end{cases}$$

So $\langle D \rangle \in L(D) \iff \langle D \rangle \notin L(D)$ ↯

<u>Conclusion</u> D cannot exist!

so H cannot exist

Matrix of TM's



Entry $\langle M_i\rangle\langle M_j\rangle = \begin{cases} 1 & \text{if } \langle M_j\rangle \in L(M_i) \\ 0 & \text{otherwise} \end{cases}$

- Suppose H exists
- Then D exists (just swapping accept and reject state)
- Hence $D = \langle M_i\rangle$ for some M_i in the list of all TM's
- But D disagrees with M_i on input $\langle M_i\rangle$

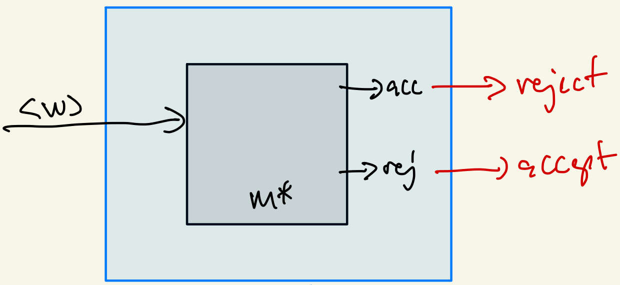
  $\langle M_i\rangle \in L(M_i) \Leftrightarrow \langle M_i\rangle \notin L(D) = \langle M_i\rangle$ ↯
- So D is not in the lists and hence does not exist $\Rightarrow$ H does not exist

$\Rightarrow A_{TM}$ is not decidable

Theorem 4.22

L is Turing-decidable
$\Updownarrow$
L and $\overline{L}$ are Turing recognizable

$\Downarrow$: Let M^* decide L. M^* always stops and $L(M^*) = L$
so M^* recognizes L and $\overline{M^*}$ recognizes $\overline{L}$:

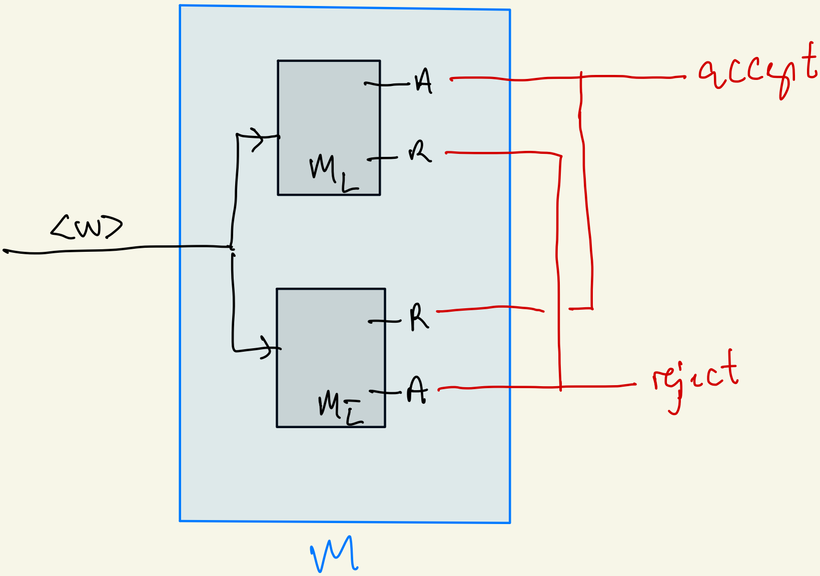


$\overline{M^*}$

$\Uparrow$: Let $M_L, M_{\overline{L}}$ recognize L respectively $\overline{L}$

Wrong approach: 1. Run M_L on w
If M_L accepts we accept w
If M_L rejects we reject w
if M_L loops .... ??

We need to run M_L and $M_{\bar{L}}$ in parallel



M : simulates each of M_L, $M_{\bar{L}}$ one step at a time (on different tapes)

as $w \in L$ or $w \in \bar{L}$ one of M_L, $M_{\bar{L}}$ will accept/reject after a finite # of steps

Theorem For every language L over universal alphab.
exactly one of the following holds

1) L and $\overline{L}$ are decidable
2) none of $L, \overline{L}$ are recognizable
3) L is recognizable but $\overline{L}$ is not recognizable or
$\overline{L}$ .. - - - L . . - - - .

$\overline{L} \backslash L$	D	R	NR
D	*	–	–
R	–	–	*
NR	–	*	*

D = decidable

R = recognizable but not decidable

NR = not recognizable

* = possible

– = not possible

Corollary $\overline{A_{TM}}$ is not recognizable

$\overline{A_{TM}} = \{ \langle w' \rangle \mid$ 1. No prefix of $\langle w' \rangle$ codes a TM
or
2. For every $\langle M \rangle$ s.t $\langle w' \rangle = \langle M \rangle \langle w \rangle$
we have $w \notin L(M)$ $\}$