# Undecidability

Zeyu Mi

*2019-12-13*


**Adapted From:**

IALC 9.1 9.2 9.3

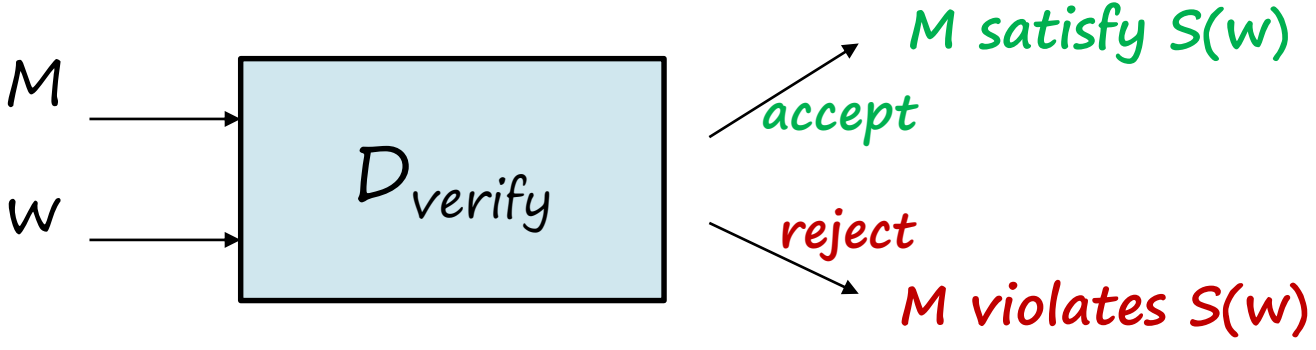
Stanford CS103 Undecidability

Universal Program Verifying

Verify the correctness of every program.

# Universal Program Verifying

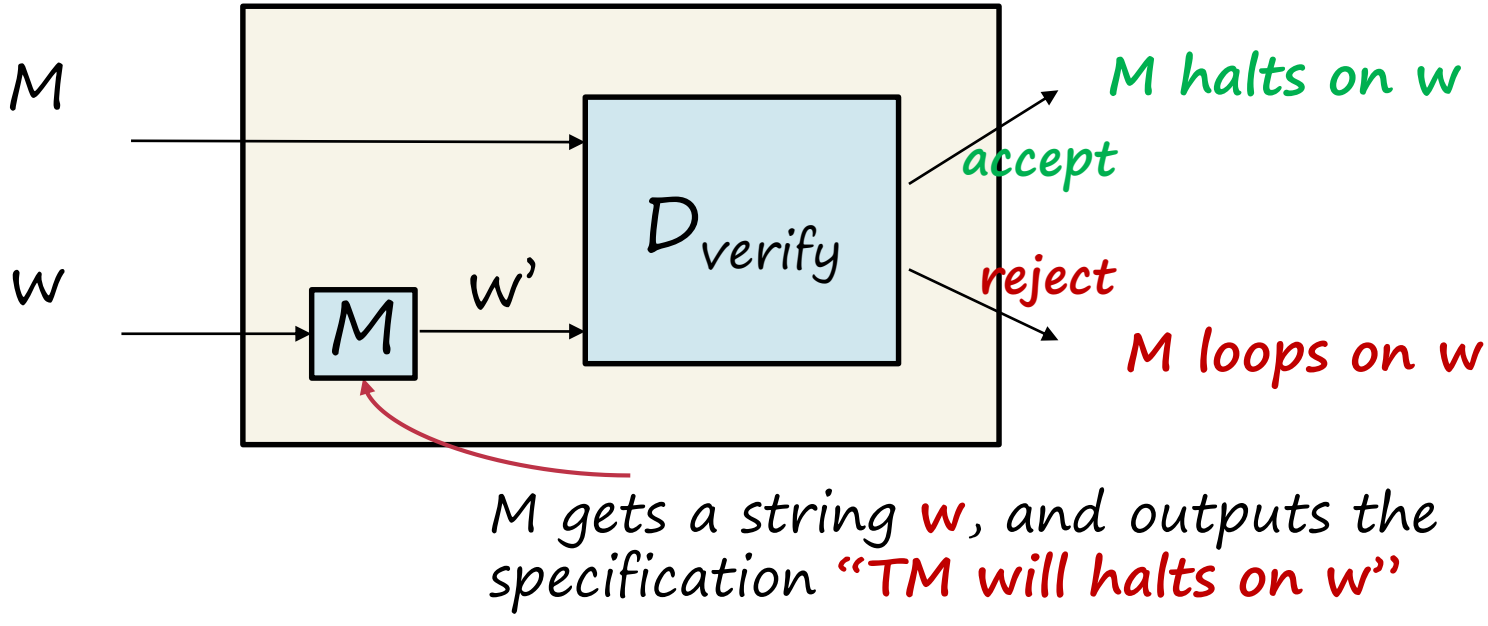
- Does there exist a TM to **decide** whether a TM satisfies a specification (some property of TM, encoded as binary string) given by an input?
- Or whether $L_{verify} = \{\langle M, w\rangle | M\ satisfies\ S(w)\}$, where $S(w)$ is the specification given by $w$, is in $\mathbf{R}$ ?



# Universal Program Verifying

- We can reduce a halting Program to this problem.
- Or if $D_{verify}$ exists, we can use it to solve the Halting Problem.

# Universal Program Verifying

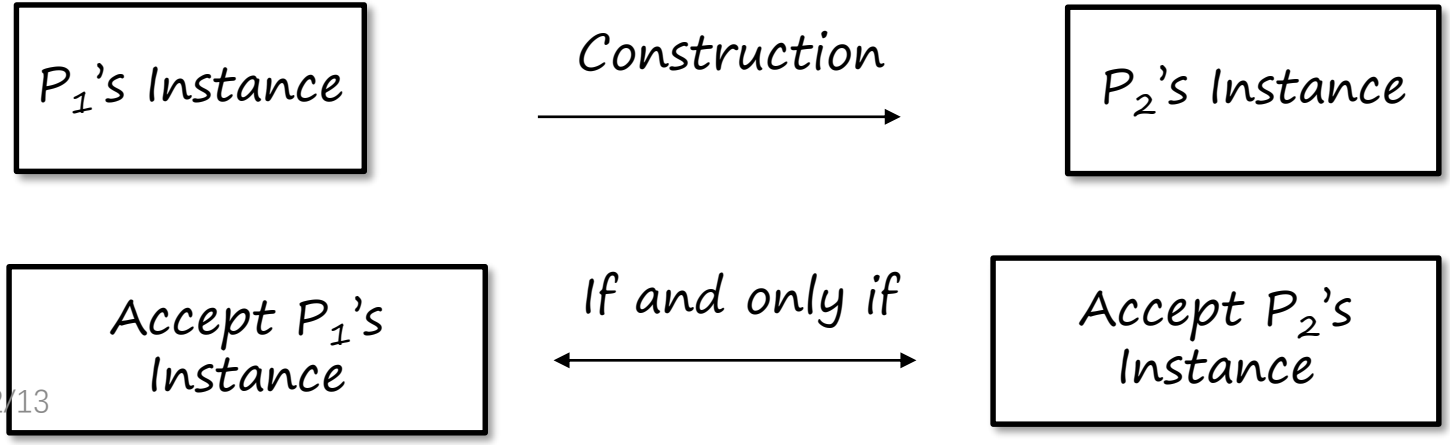


# Universal Program Verifying

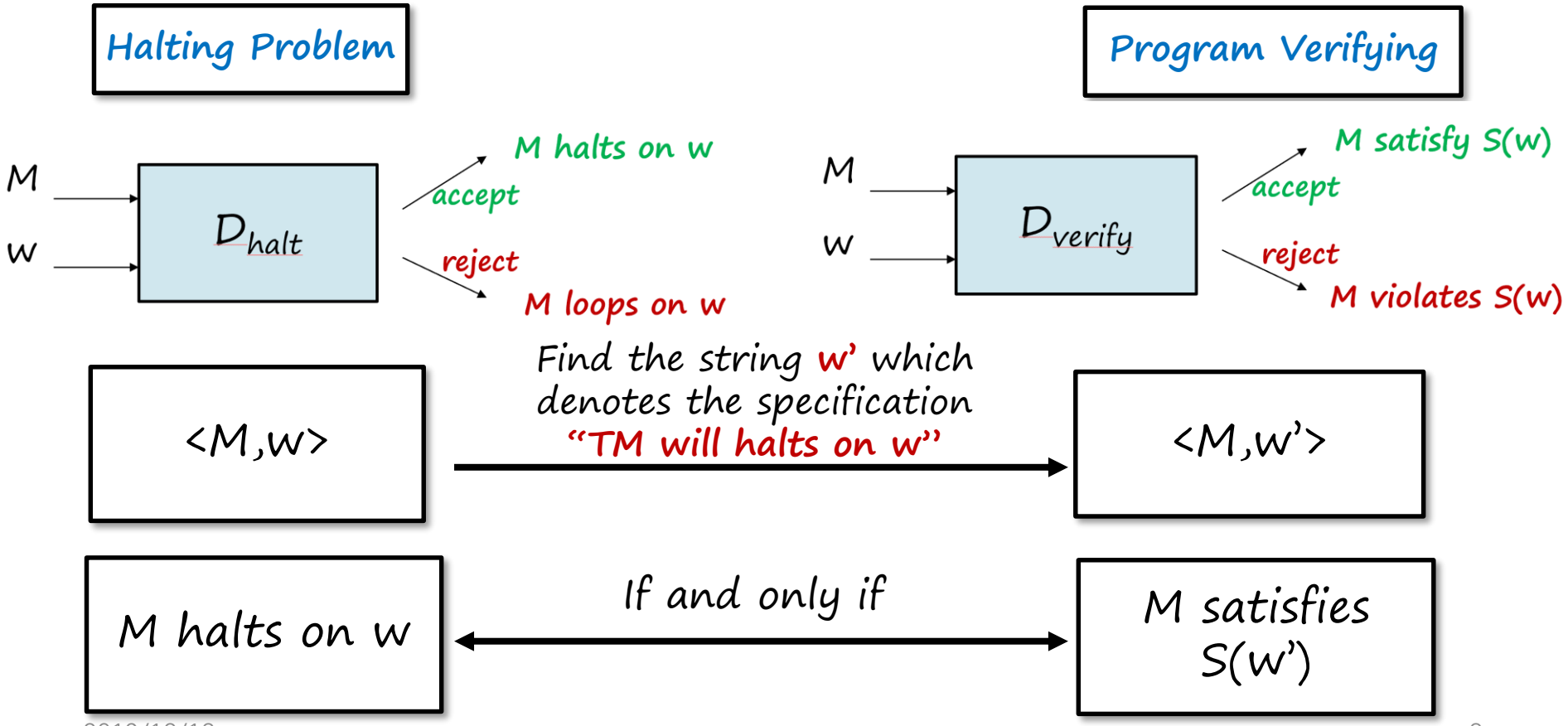
- We can reduce a halting program to this problem.
- Or if $D_{verify}$ exists, we can use it to solve the halting problem.
- But we've proved halting problem is undecidable.
- So $D_{verify}$ does not exists.
- **Universal Program Verifying is undecidable.**

# Reduction

- We prove that universal program verifying problem is undecidable by reducing the halting problem to it.
- Reducing from problem $P_1$ to problem $P_2$ means:
  - An algorithm to convert $P_1$'s instance to $P_2$'s instance.
  - $P_1$'s instance is accepted in $P_1$ if and only if its converted $P_2$'s instance is accepted in $P_2$



# Reductions


Halting Problem
Program Verifying
M
w
$D_{halt}$
M halts on w
accept
reject
M loops on w
M
w
$D_{verify}$
M satisfy S(w)
accept
reject
M violates S(w)
<M,w>
Find the string w' which denotes the specification "TM will halts on w"
<M,w'>
M halts on w
If and only if
M satisfies S(w')

# Reductions

- If we can reduce a problem $P_1$ to another problem $P_2$, we intend to use the solution of $P_2$ to solve $P_1$, which means that $P_2$ is "harder" than $P_1$.
  - **If $P_1$ is undecidable then so is $P_2$.**
  - **If $P_1$ is non-RE, then so is $P_2$.**
- So if we want to prove some problem $P$ is undecidable, we only need to reduce some already known undecidable problem to $P$
  - $L_u$, $L_{halt}$, etc.

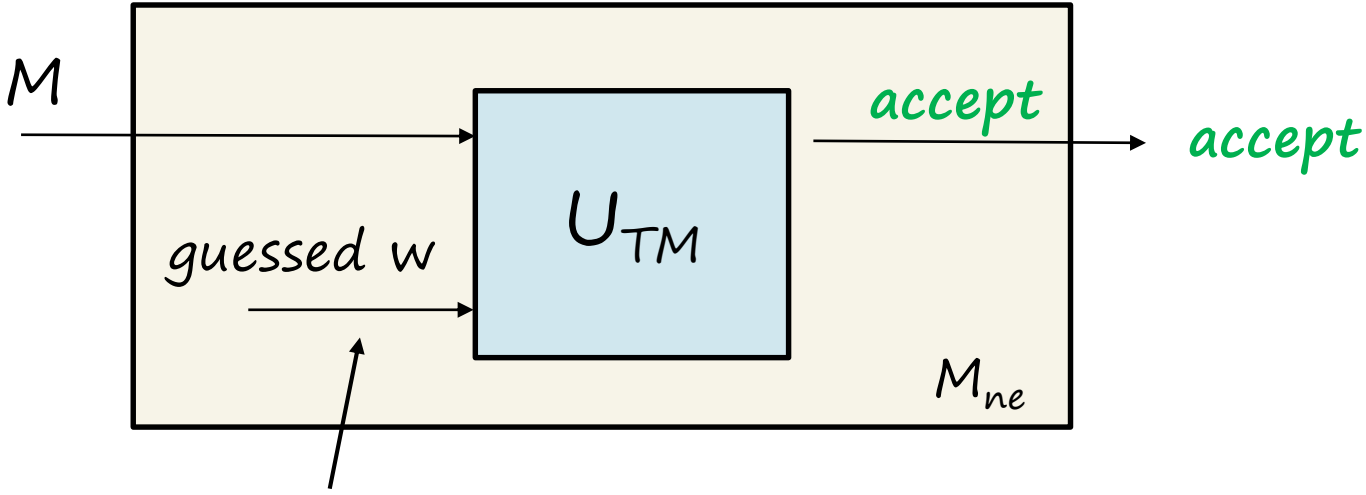
# TM that Accepts the Empty Language

- As an example of reductions involving TMs, let us investigate two languages called $L_e$ and $L_{ne}$.
  - If $L(M) = \emptyset$, that is, $M$ does not accept any input, then the code of $M$ is in $L_e$
  - Otherwise, if $L(M)$ is not the empty language, then the code of $M$ is in $L_{ne}$

$$L_e = \{M | L(M) = \emptyset\}$$

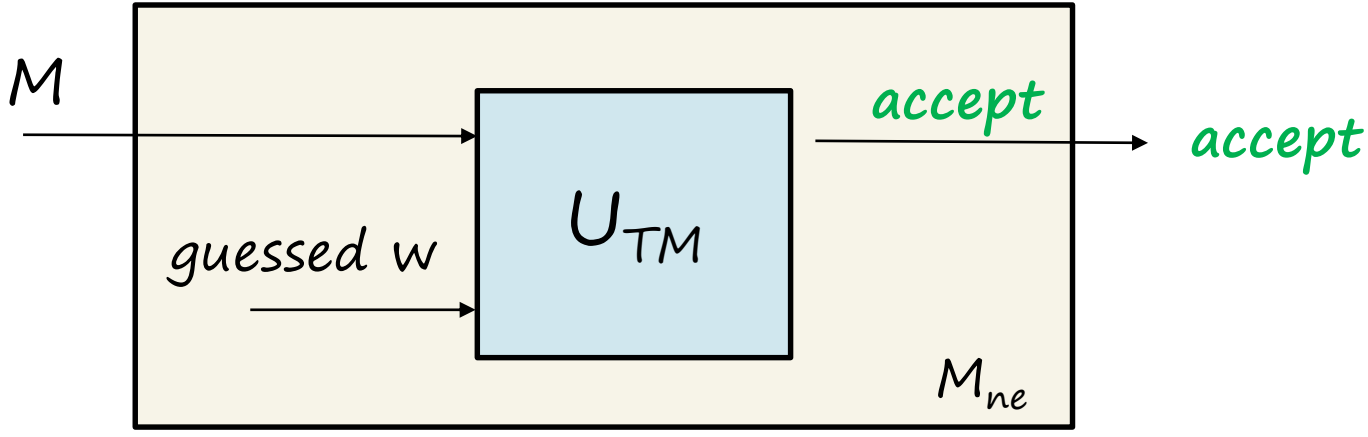
$$L_{ne} = \{M | L(M) \neq \emptyset\}$$

# $L_{ne} \in$ RE




We guess all the w simultaneously instead of guessing w one by one in case getting into a forever loop (For more information, ref to 8.4.4 Nondeterministic TM)

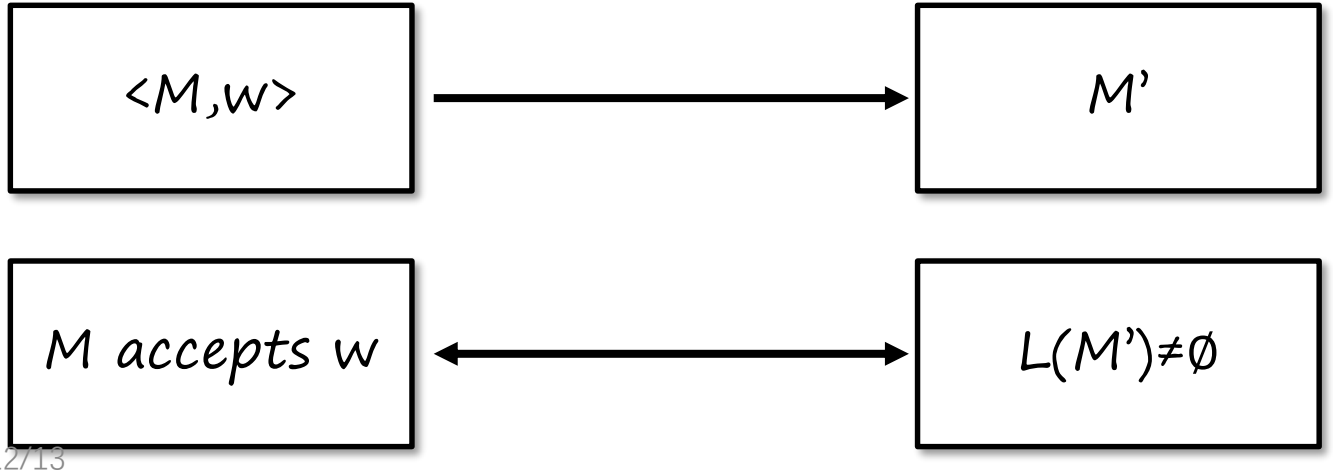
# $L_{ne} \in \mathbf{RE}$



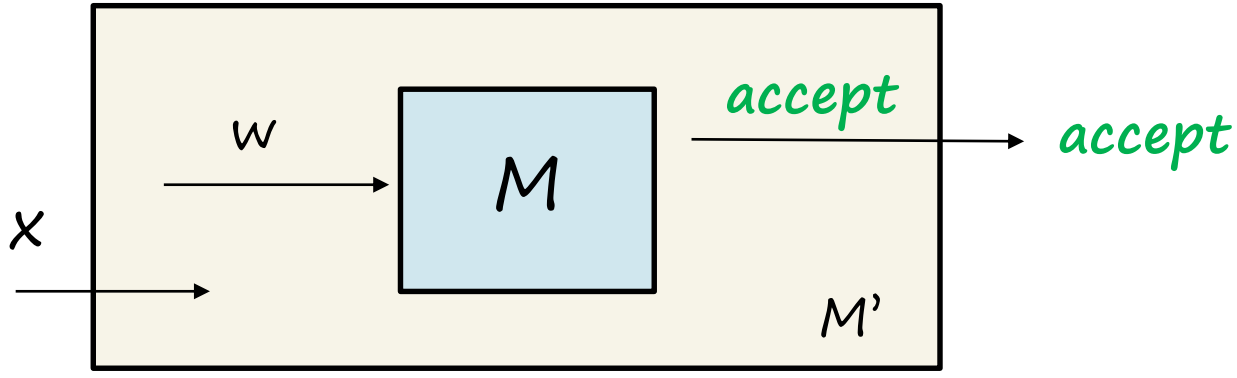

- If $M \in L_{ne}$, which means $M$ accepts some string $w$, we'll finally guess $w$, then $M_{ne}$ accepts $M$.
- If $M \notin L_{ne}$, which means $M$ accepts no string, then we will guess a string that does not exist forever, then $M_{ne}$ does not accept $M$.

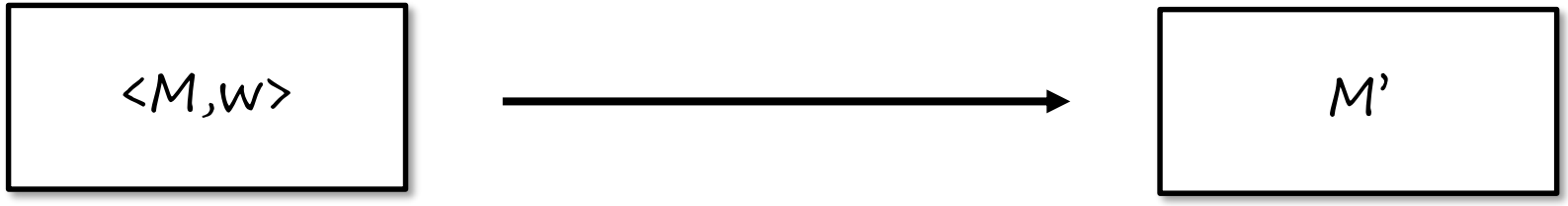
# $L_{ne} \notin \mathbf{R}$

- We'll prove this by reducing $L_u$ to $L_{ne}$.
- We need to design an algorithm that converts an input that is a pair $\langle \boldsymbol{M}, \boldsymbol{w} \rangle$ into a TM $\boldsymbol{M'}$ such that $\boldsymbol{L}(\boldsymbol{M'}) \neq \emptyset$ if and only if $\boldsymbol{M}$ accepts the input $\boldsymbol{w}$.

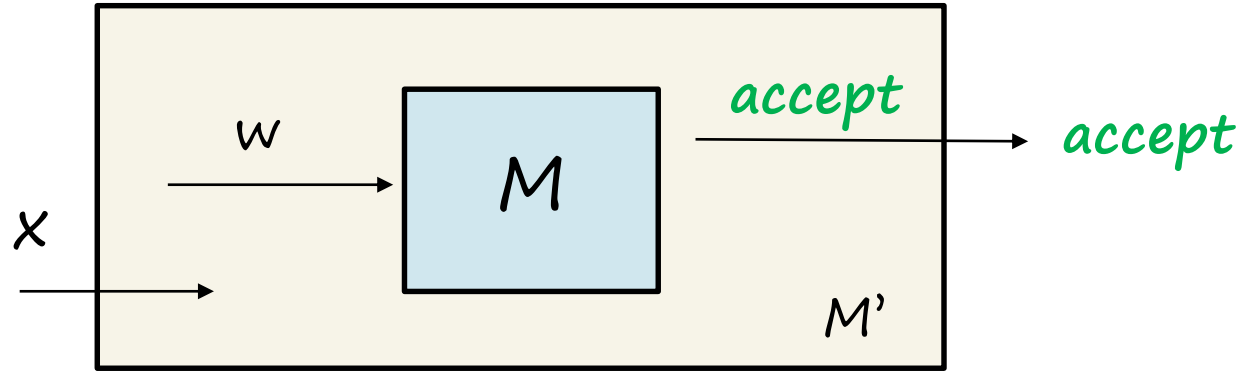


# $L_u \rightarrow L_{ne}$



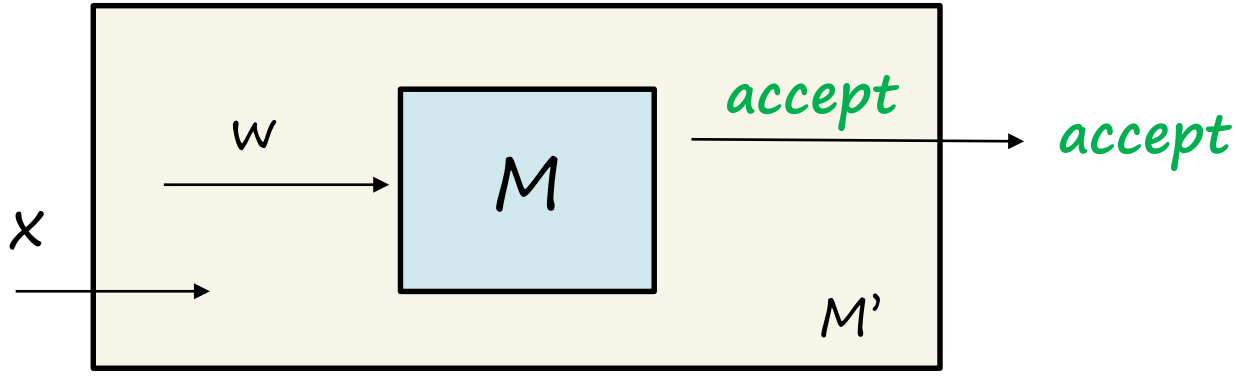



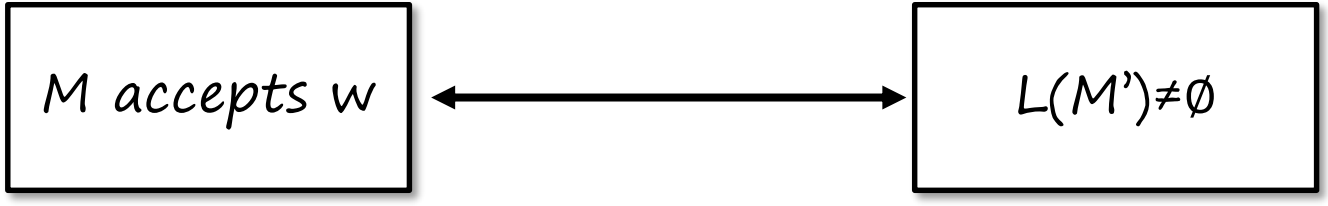
# $L_u \rightarrow L_{ne}$




- If $M$ accepts $w$
  - Then $M'$ accepts any strings, which means $L(M') \neq \emptyset$
- If $L(M') \neq \emptyset$
  - Then $M$ must accept $w$

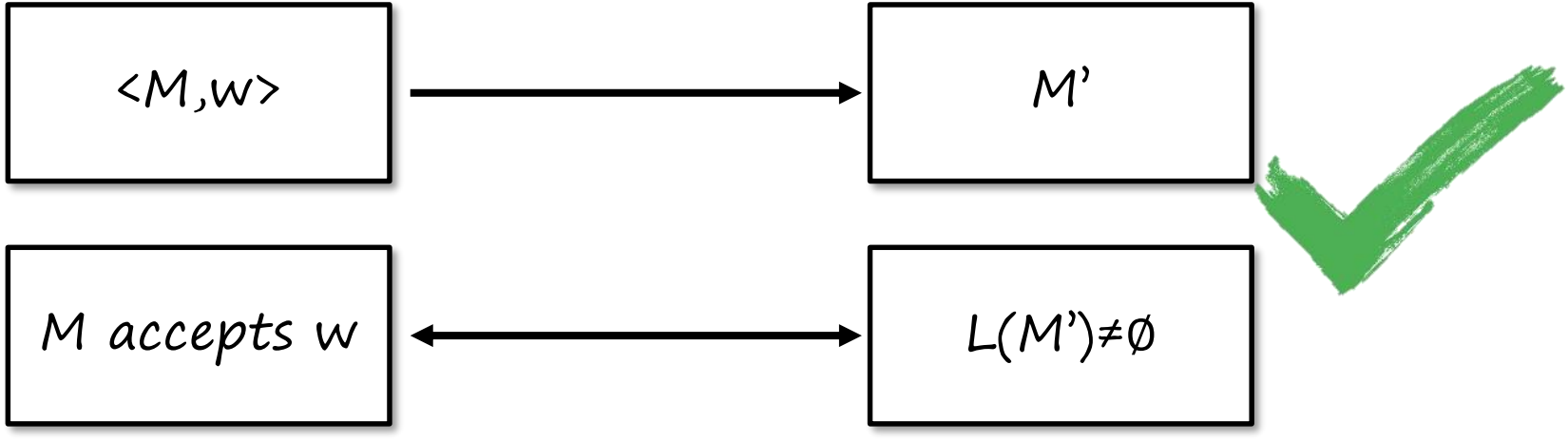
# $L_u \rightarrow L_{ne}$



# $L_{ne} \notin \mathbf{R}$

- We'll prove this by reduce $L_u$ to $L_{ne}$.
- We need to design an algorithm that converts an input that is a pair $\langle \boldsymbol{M}, \boldsymbol{w} \rangle$ into a TM $\boldsymbol{M'}$ such that $\boldsymbol{L}(\boldsymbol{M'}) \neq \emptyset$ if and only if $\boldsymbol{M}$ accepts input $\boldsymbol{w}$.

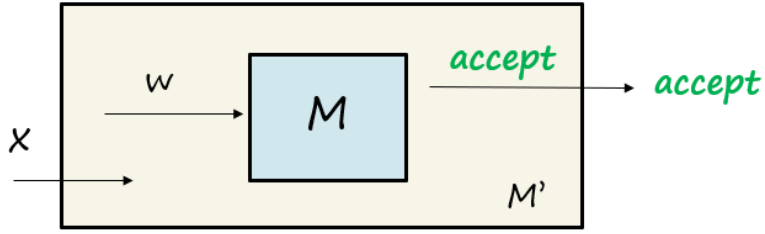


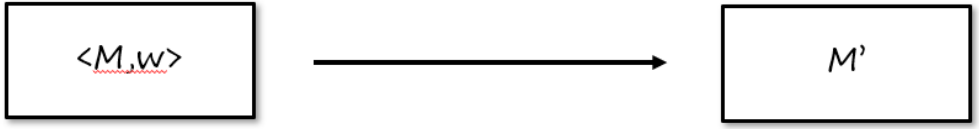
# TM that Accepts the Empty Language

- We now know that $L_{ne} \in \mathbf{RE}$ and $L_{ne} \notin \mathbf{R}$.
- As $L_e = \overline{L_{ne}}$, so if $L_e \in \mathbf{RE}$, then $L_{ne} \in \mathbf{R}$. (why?)
- But $L_{ne} \notin \mathbf{R}$. So we have $\boldsymbol{L_e \notin \mathbf{RE}}$

# L1规约到L2

- 规约中构造是针对两个问题（语言）的**输入**。
- 举例: $L_u \rightarrow L_{ne}$
  - $L_u$的输入为图灵机加字符串$\langle M, w\rangle$, $L_{ne}$的输入为一个图灵机。
  - 所以第一步构造就是将一个**图灵机加字符串**转换为一个**图灵机.**



# L1规约到L2

- 我们设对于任意L1的输入$s$，构造转换得到的L2的输入为$f(s)$.
- 我们下面需要说明

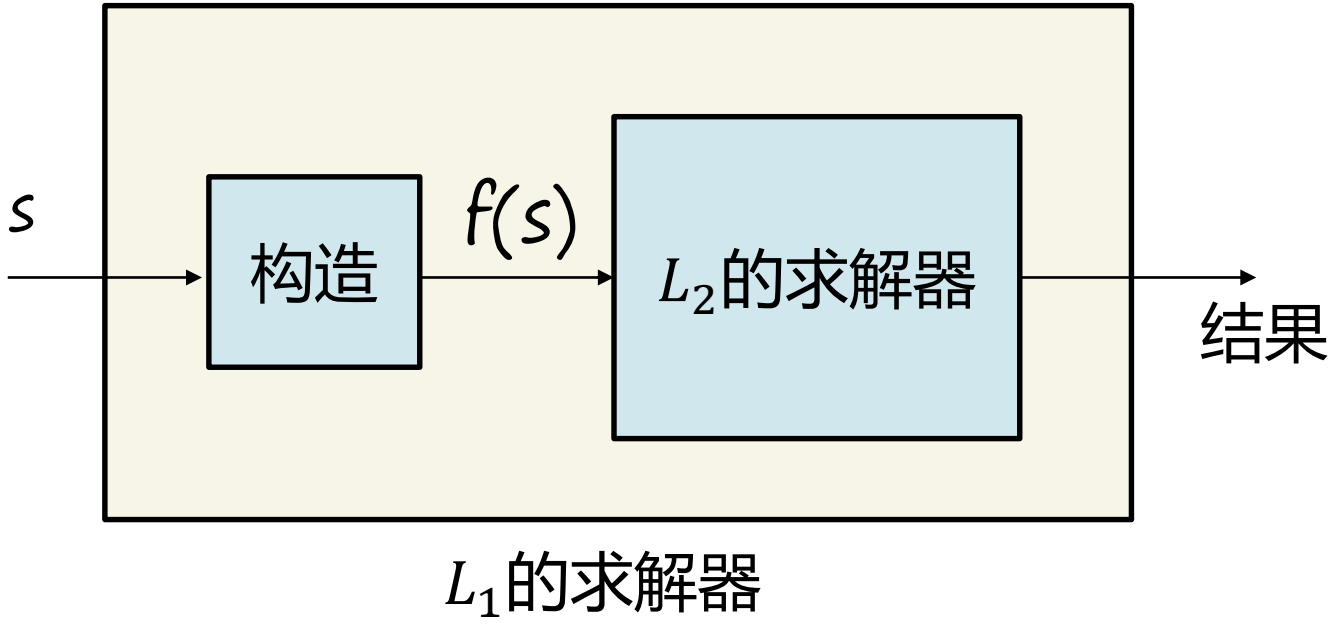
$$\boldsymbol{s \in L_1 \Leftrightarrow f(s) \in L_2}$$

- 也就是对于任意输入，
  - 如果这个输入在$L_1$中，那么转换后的输入应该在$L_2$中
  - 如果这个输入不在$L_1$中，那么转换后的也不应该在$L_2$中

# L1规约到L2

If $L_1$ is undecidable then so is $L_2$.

If $L_1$ is non-RE, then so is $L_2$.



# 将乘法规约到加法

$$L_1 = \{(a, b, P) | a * b = P\} \qquad L_2 = \{(a_1, a_2, \dots a_n, S) | a_1 + a_2 + \cdots a_n = S\}$$

1. 构造，将$L_1$的一个任一输入转换为$L_2$的输入

$$(a, b, P) \longrightarrow (\underbrace{a, a, \dots a}_{\text{b个}a}, P)$$

2. 证明：$(a, b, P) \in L_1$ 当且仅当 $(\overbrace{a, a, \dots a}^{\text{b个}a}, P) \in L_2$

$$a * b = P \longleftrightarrow a + a + .. a = P$$

$$\boldsymbol{L_{eq} = \{\langle M_1, M_2\rangle | L(M_1) = L(M_2)\}}$$

- $L_e = \{M | L(M) = \emptyset\}$

# $L_{eq} = \{\langle M_1, M_2\rangle | L(M_1) = L(M_2)\}$

- $L_e = \{M | L(M) = \emptyset\}$

1. 构造，将$L_e$的一个任一输入转换为$L_{eq}$的输入

$$M \qquad\qquad \langle M, M_1\rangle \ \ (L(M_1) = \emptyset)$$

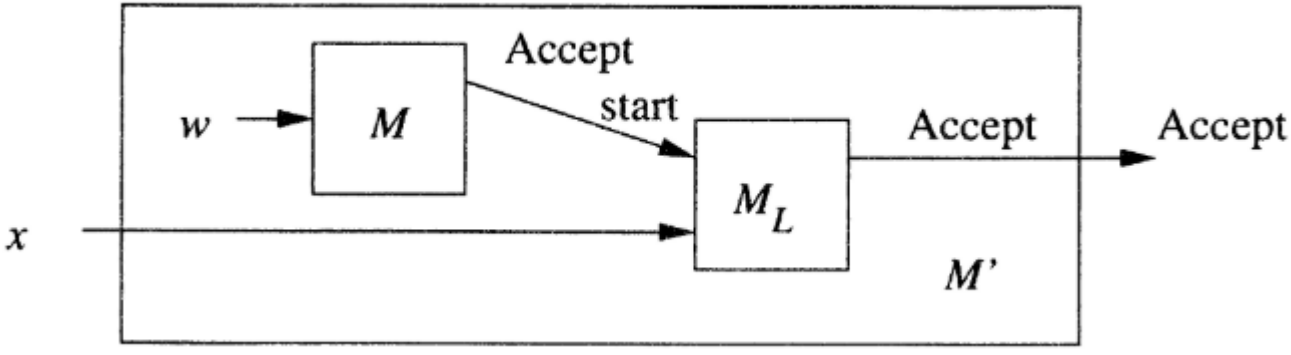
2. 证明，$M \in L_e$当且仅当 $\langle M, M_1\rangle \in L_{eq}$
   - 如果$M \in L_e$, 那么$L(M) = \emptyset$, 有$L(M) = L(M_1)$, 所以$\langle M, M_1\rangle \in L_{eq}$
   - 如果$\langle M, M_1\rangle \in L_{eq}$，那么$L(M) = L(M_1) = \emptyset$, 所以$M \in L_e$

# Rice's Theorem

- $\mathcal{P}$: set of RE languages （串集合的集合）
- *Non-trivial:* $\mathcal{P} \neq \emptyset$ ， $\mathcal{P} \neq \{all\ re\ languages\}$
- $L_{\mathcal{P}} = \{M | L(M) \in \mathcal{P}\}$ 不可判定
- $L_u$ 规约到 $L_{\mathcal{P}}$
  - 假设$\emptyset \notin \mathcal{P}$, 那么至少存在一个RE语言$L \neq \emptyset \in \mathcal{P}$, 设其对应的图灵机为$M_L$

# Rice's Theorem

1. 构造。$\langle M, w\rangle$ 到 $M'$




2. 证明。$\langle M, w\rangle \in L_u$ 当且仅当 $M' \in L_{\mathcal{P}}$

   - 如果$\langle M, w\rangle \in L_u$, 有$M$ 接受$w$，此时$L(M') = L(M_L) = L \in \mathcal{P}$,所以有$M' \in L_{\mathcal{P}}$
   - 若果$\langle M, w\rangle \notin L_u$, 此时$M'$不接受任意输入，所以$L(M') = \emptyset \notin \mathcal{P}$, 所以有$M' \notin L_{\mathcal{P}}$