# Decidability

Linz 6th, Chapter 12: Limits of Algorithmic Computation, page 309ff

A property P of strings is said to be *decidable* if the set of all strings having property P is a recursive set; that is, if there is a total Turing machine that accepts input strings that have property P and rejects those that do not.

$$P \text{ is decidable} \iff \{x \mid P(x)\} \text{ is recursive.}$$

$$A \text{ is recursive} \iff \text{"}x \in A\text{" is decidable,}$$

Kozen

# Consider problems with answer YES or NO

Examples:

- Does Machine $M$ have three states ?

- Is string $w$ a binary number?

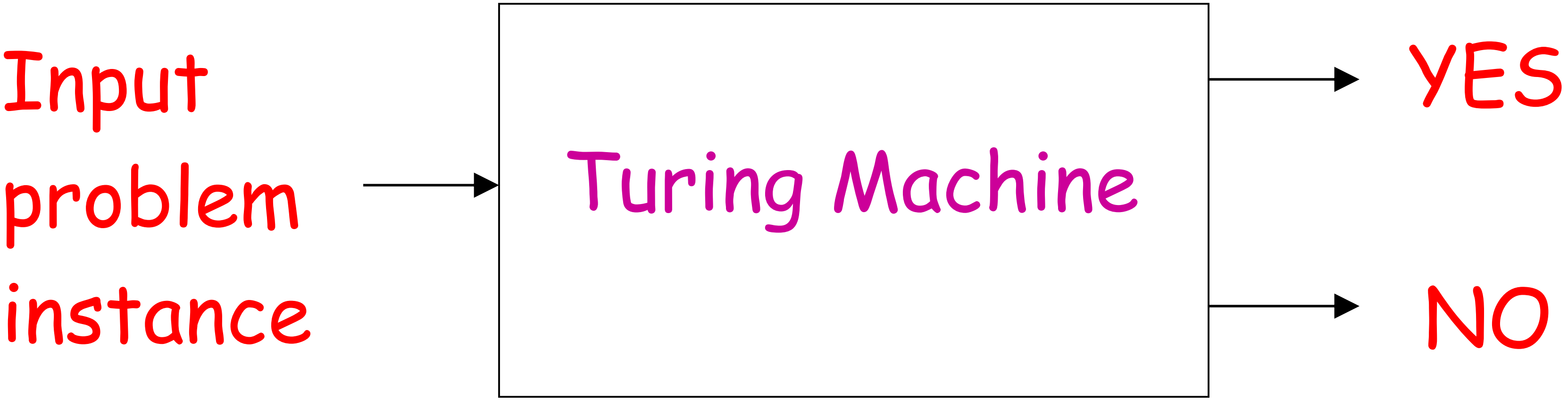
- Does DFA $M$ accept any input?

A problem is decidable if some Turing machine Solves (decides) the problem

Decidable problems:

- Does Machine $M$ have three states ?
- Is string $w$ a binary number?
- Does DFA $M$ accept any input?

The Turing machine that solves a problem answers YES or NO for each instance



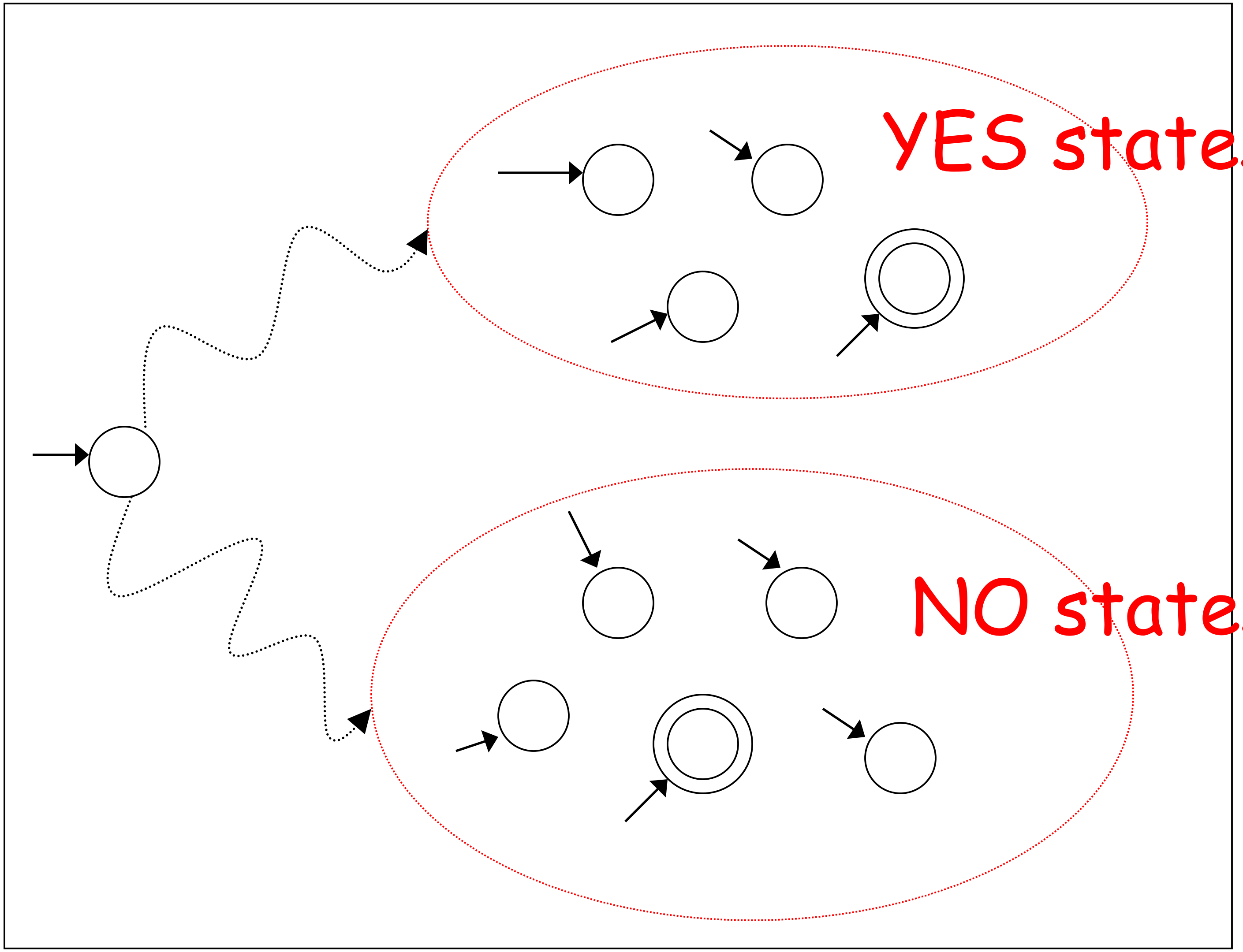
## The machine that decides a problem:

- If the answer is YES
  then halts in a <u>yes state</u>

- If the answer is NO
  then halts in a <u>no state</u>

These states may not be final states

# Turing Machine that decides a problem




YES and NO states are halting states

# Difference between
# Recursive Languages and Decidable problems

For decidable problems:

The YES states may not be final states

[What does the author mean?]

No harm in assuming YES states are final.

Decidable = Recursive

## Some problems are undecidable:

which means:
there is no Turing Machine that
solves all instances of the problem

## A simple undecidable problem:

The membership problem

# The Membership Problem

Input: • Turing Machine $M$, and

• String $w$

Question: Does $M$ accept $w$ ?

$$w \in L(M) ?$$

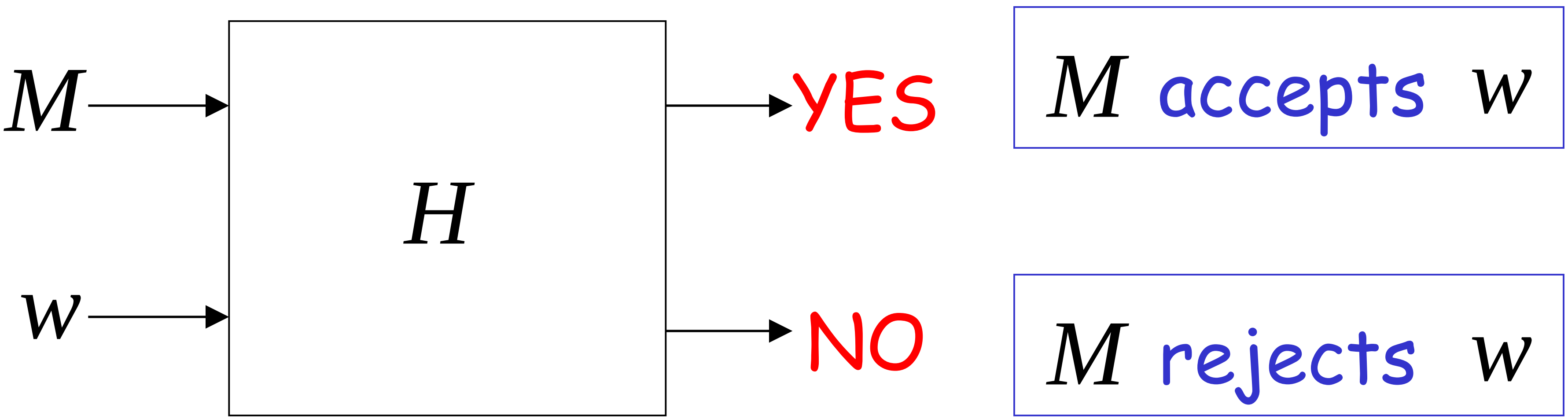
Theorem:

The membership problem is undecidable

(there are $M$ and $w$ for which we cannot decide whether $w \in L(M)$ )

Proof: Assume for contradiction that the membership problem is decidable

Assume there exists a Turing Machine $H$ that decides/solves the membership problem



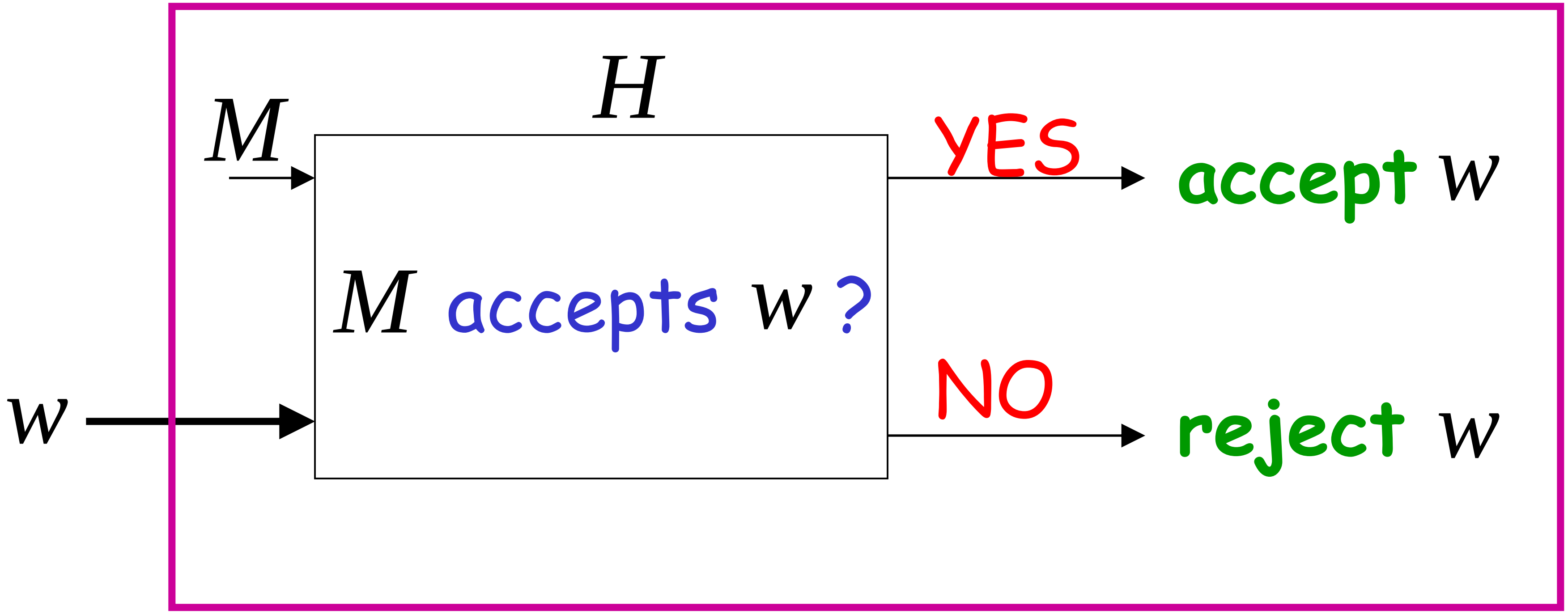
Let $L$ be a recursively enumerable language

Let $M$ be the Turing Machine that accepts $L$

We will prove that $L$ is also recursive:

we will describe a Turing machine that accepts $L$ and halts on any input

Turing Machine that accepts $L$
and halts on any input



Therefore, $L$ is recursive

Since $L$ is chosen arbitrarily, we have proven that every recursively enumerable language is also recursive

But there are recursively enumerable languages which are not recursive

**Contradiction!!!!**

Therefore, the membership problem is undecidable

END OF PROOF

## Another famous undecidable problem:

The halting problem

# ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

*By* A. M. TURING.

[Received 28 May, 1936.—Read 12 November, 1936.]

The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. Although the subject of this paper is ostensibly the computable *numbers*,

# The Halting Problem

Input: • Turing Machine $M$, and

• String $w$

Question: Does $M$ halt on input $w$ ?

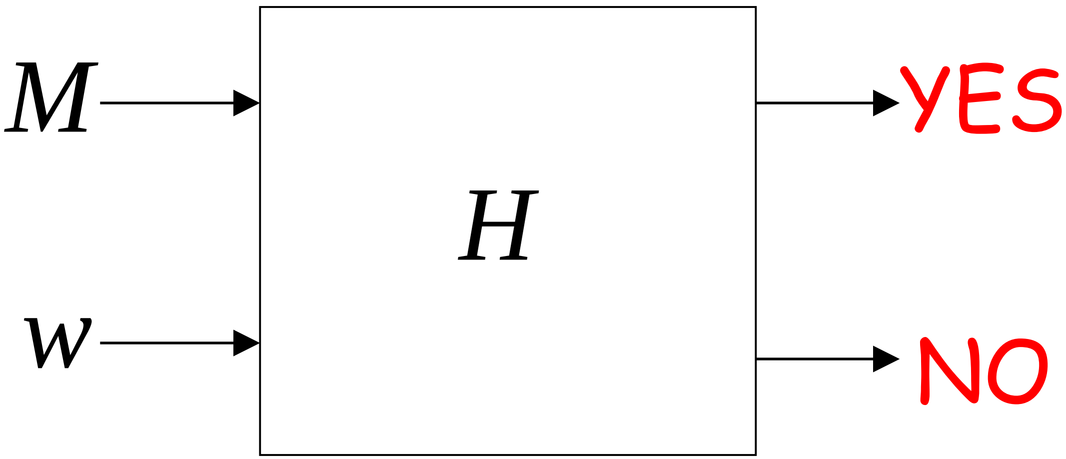
Theorem:

The halting problem is undecidable

(there are $M$ and $w$ for which we cannot decide whether $M$ halts on input $w$ )

Proof: Assume for contradiction that the halting problem is decidable

Thus, there exists Turing Machine $H$
that solves the halting problem




$M$ halts on $w$

$M$ doesn't halt on $w$

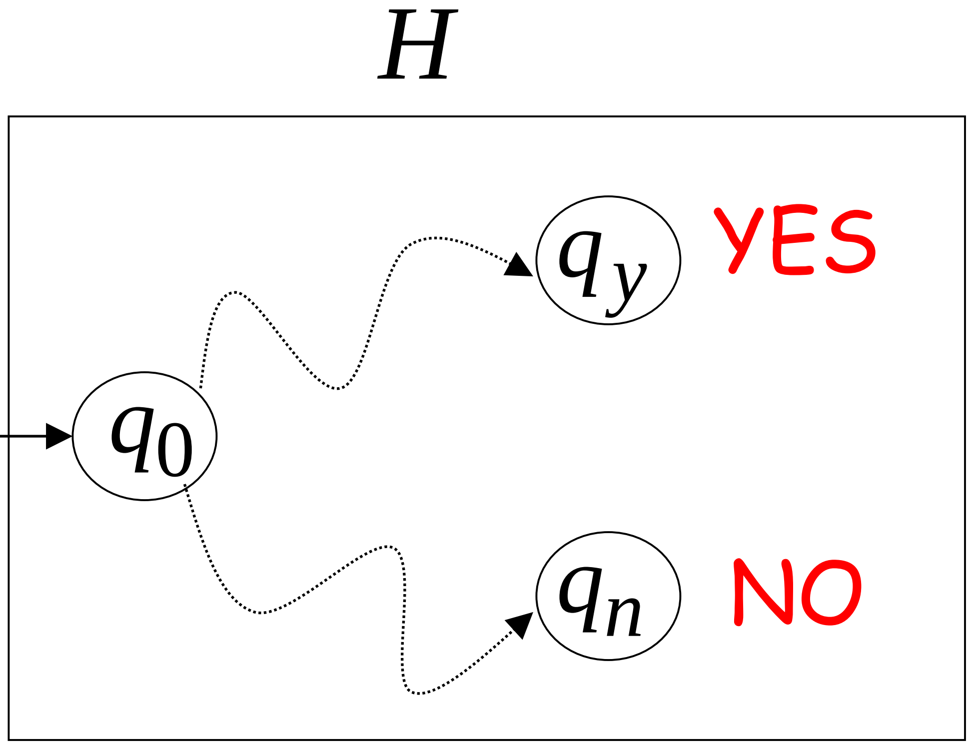
# Construction of $H$

Input:
initial tape contents

$w_M \; w$

Encoding of $M$

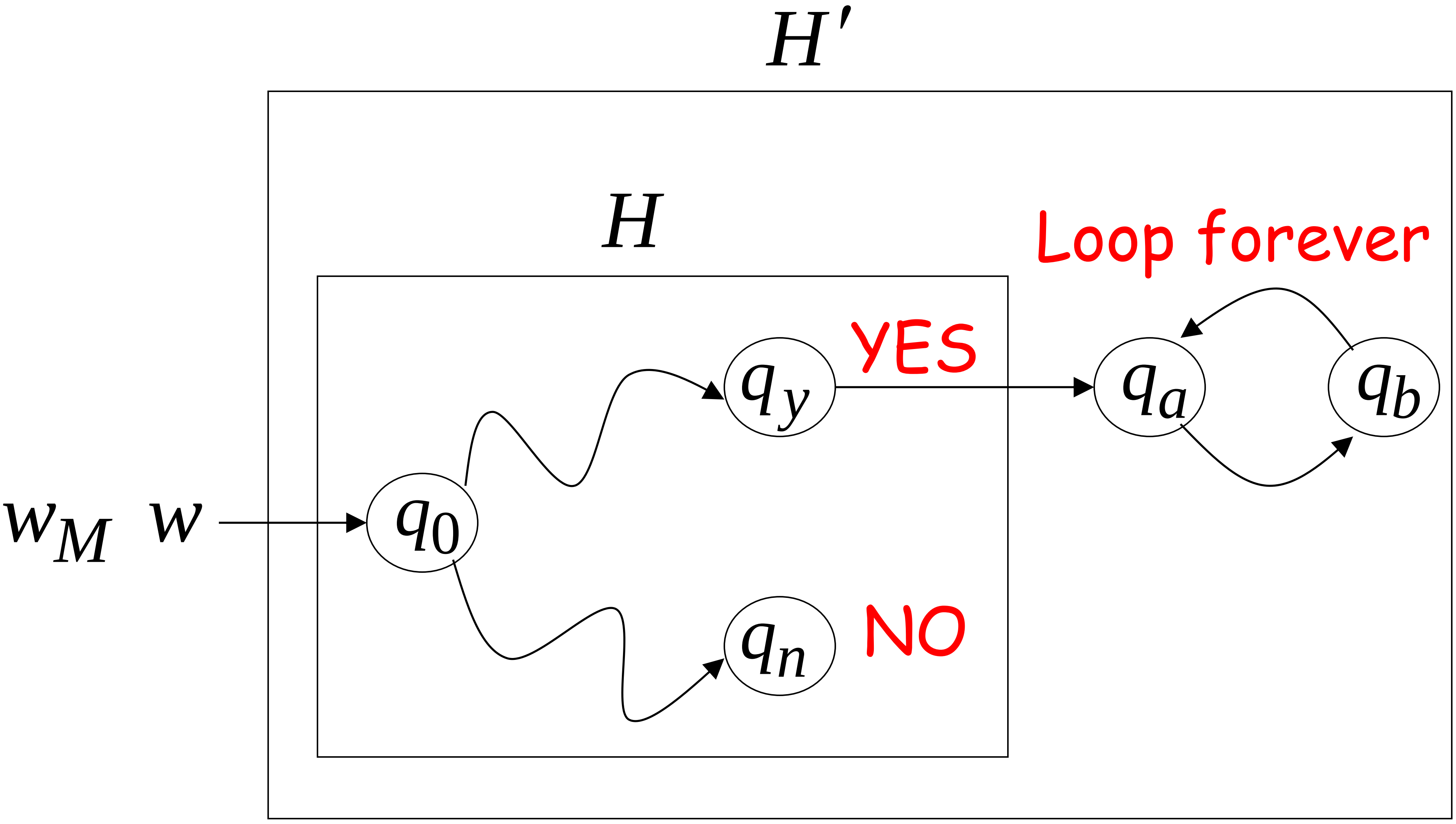
String $w$



Construct machine $H'$ :

If $H$ returns YES then loop forever

If $H$ returns NO then halt



$H'$
$H$
Loop forever
YES
NO
$q_y$
$q_a$
$q_b$
$w_M \ w$
$q_0$
$q_n$

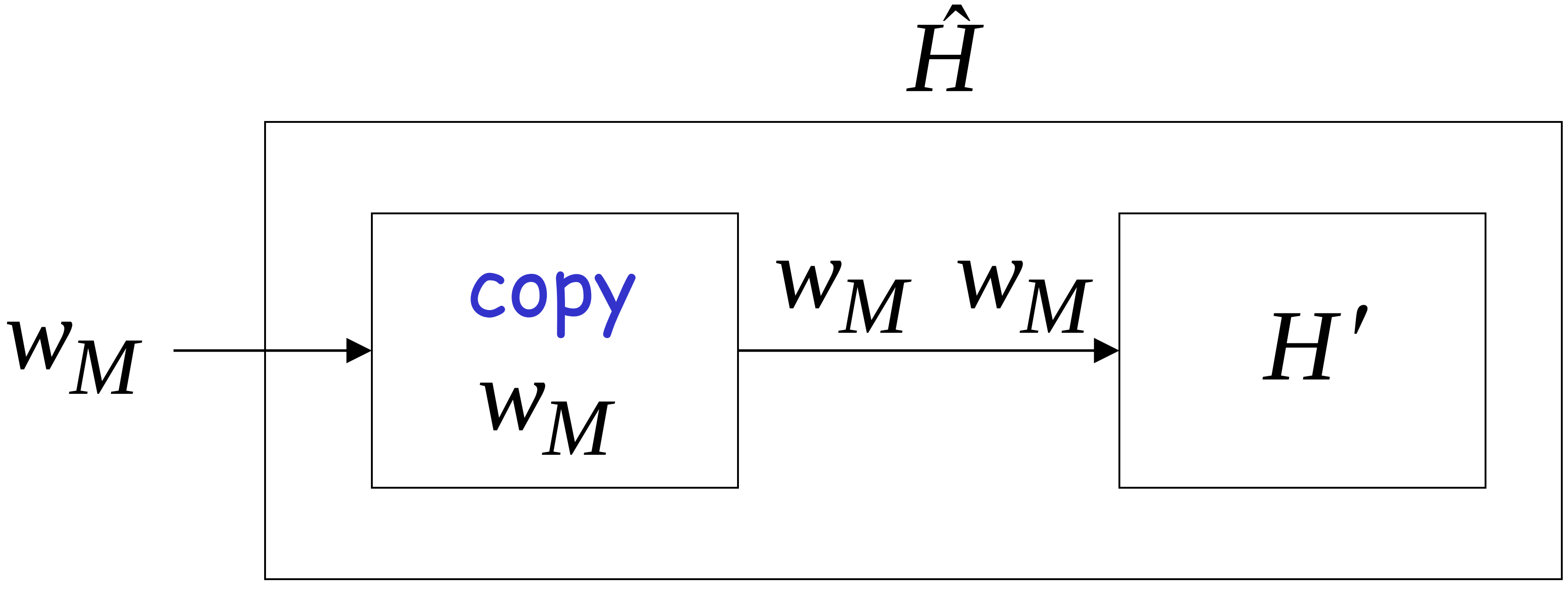
Construct machine $\hat{H}$ :

Input: $w_M$ (machine $M$ )

If $M$ halts on input $w_M$

<u>**Then**</u> loop forever

<u>**Else**</u> halt


$\hat{H}$
$w_M$
copy
$w_M$
$w_M$
$w_M$
$H'$

Run machine $\hat{H}$ with input itself:

Input: $w_{\hat{H}}$ (machine $\hat{H}$ )

If $\hat{H}$ halts on input $w_{\hat{H}}$

<u>**Then**</u> loop forever

<u>**Else**</u> halt

$\hat{H}$ on input $w_{\hat{H}}$ :

If $\hat{H}$ halts then loops forever

If $\hat{H}$ doesn't halt then it halts

NONSENSE !!!!!

Therefore, we have contradiction

The halting problem is undecidable

END OF PROOF

## Another proof of the same theorem:

If the halting problem were decidable then every recursively enumerable language would be recursive.

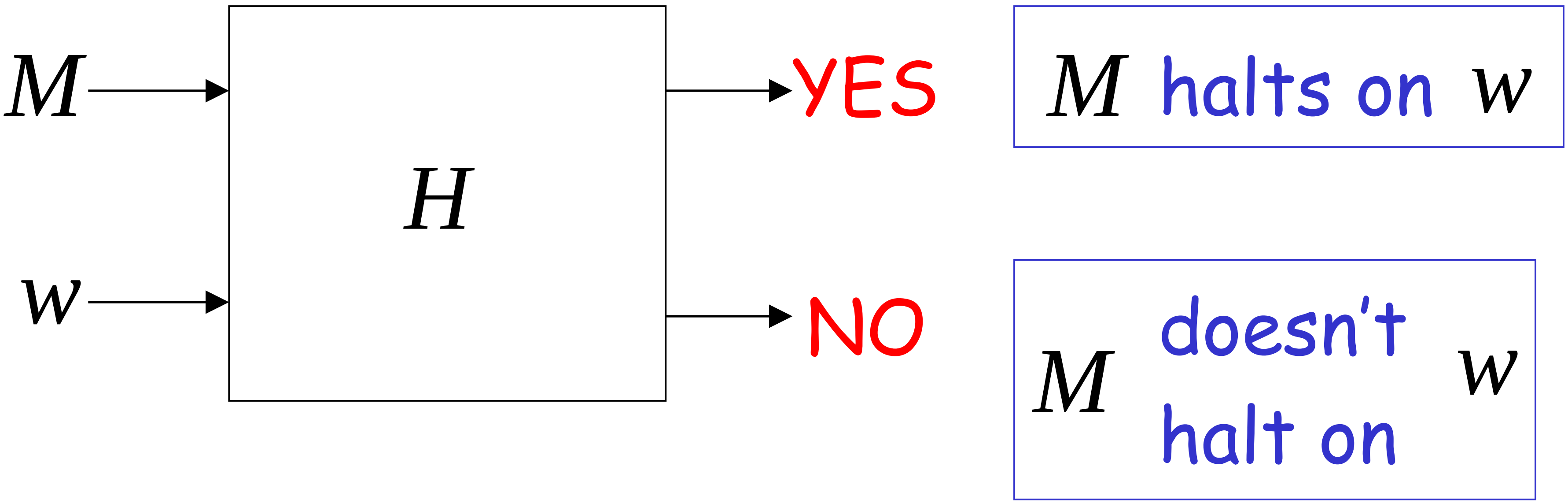
If L={M_w#w | M_w halts on w} were recursive, then every r.e. set is recursive.

**Theorem:**

The halting problem is undecidable

**Proof:** Assume for contradiction that the halting problem is decidable

## Thus, there exists Turing Machine $H$ that solves the halting problem

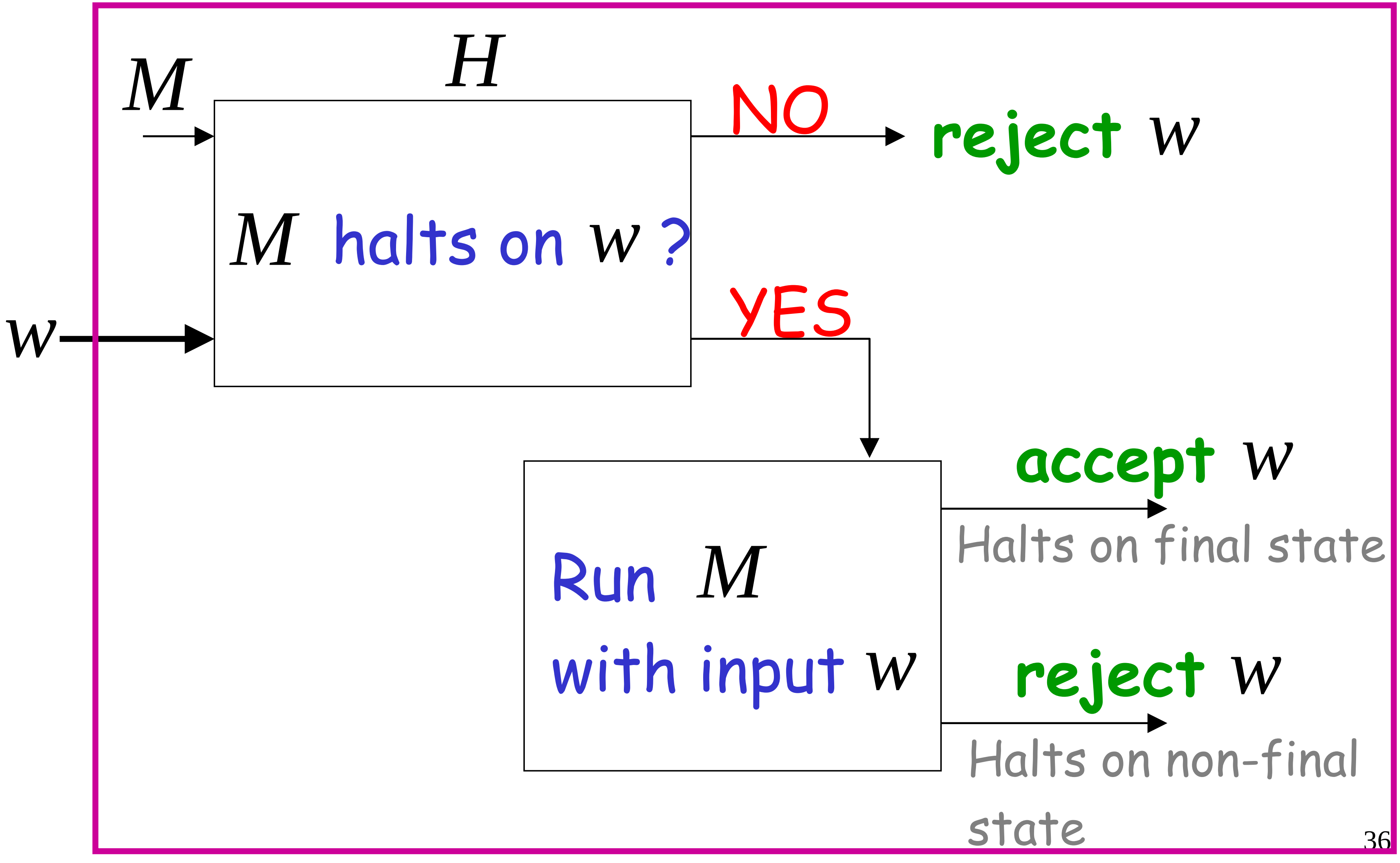


Let $L$ be a recursively enumerable language

Let $M$ be the Turing Machine that accepts $L$

we will describe a Turing machine that accepts $L$ and halts on any input, proving that $L$ is also recursive.

# Turing Machine that accepts $L$ and halts on any input



Therefore $L$ is recursive

Since $L$ is chosen arbitrarily, we have proven that every recursively enumerable language is also recursive

But there are recursively enumerable languages which are not recursive

**Contradiction!!!!**

Therefore, the halting problem is undecidable

END OF PROOF