# Undecidable problems for Recursively enumerable languages

continued…

Take a recursively enumerable language $L$

Decision problems:

- $L$ is empty?
- $L$ is finite?
- $L$ contains two different strings of the same length?

All these problems are undecidable

**Theorem:**

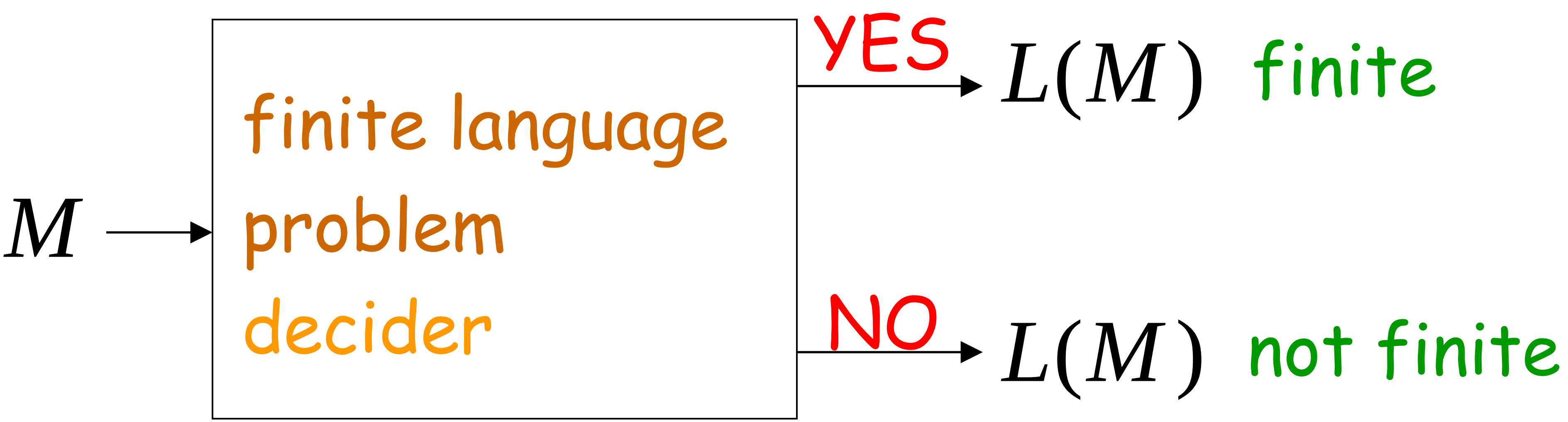
For a recursively enumerable language $L$ it is undecidable to determine whether $L$ is finite

**Proof:**

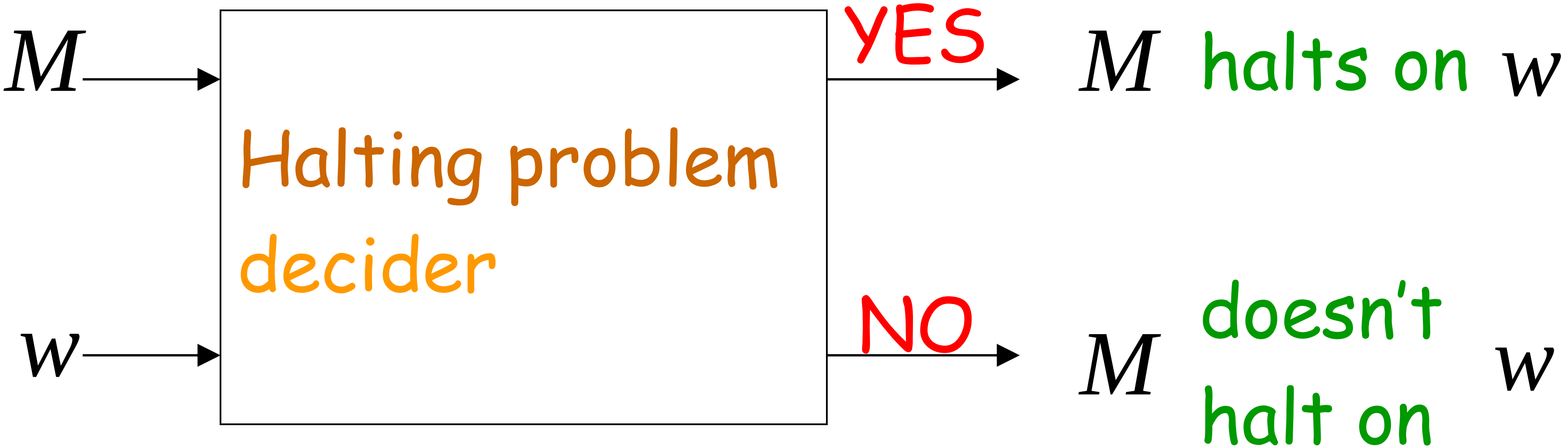
We will reduce the halting problem to this problem

Let $M$ be the TM with $L(M) = L$

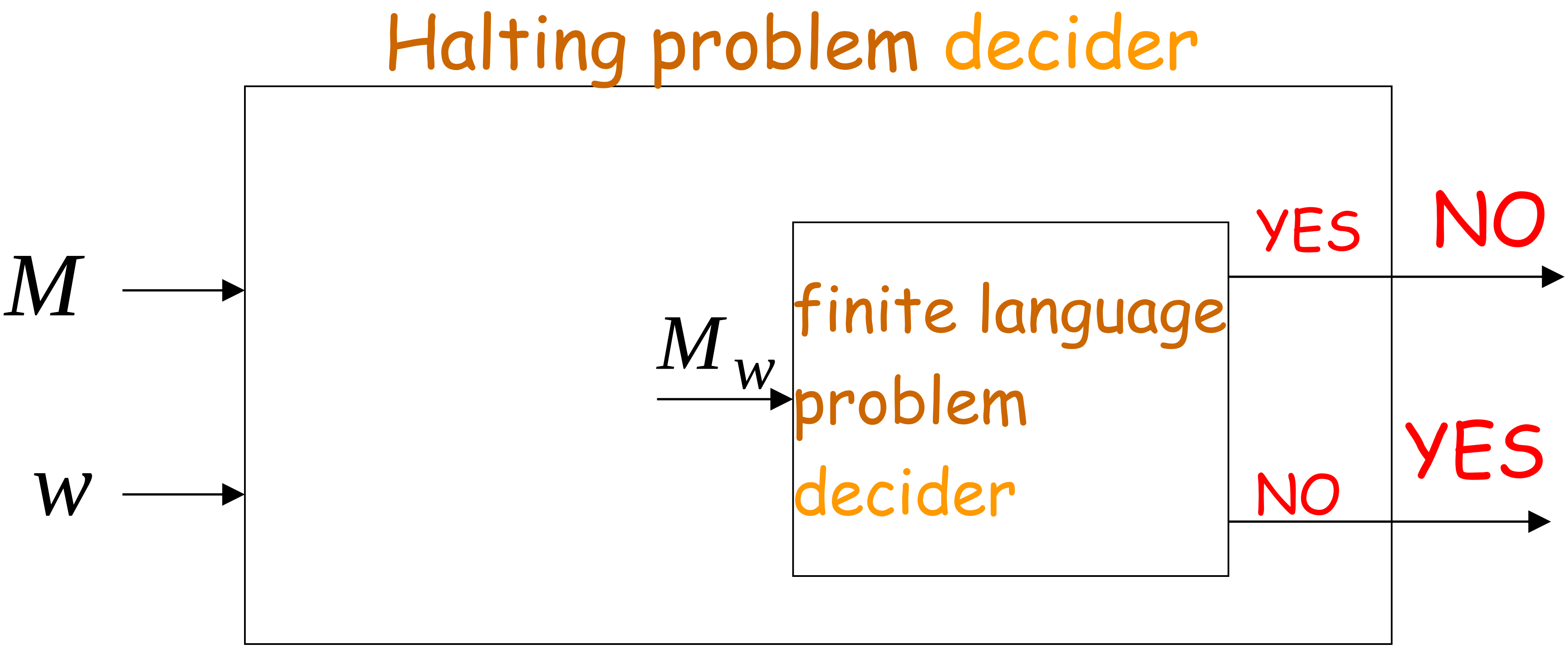
Suppose we have a decider for the finite language problem:



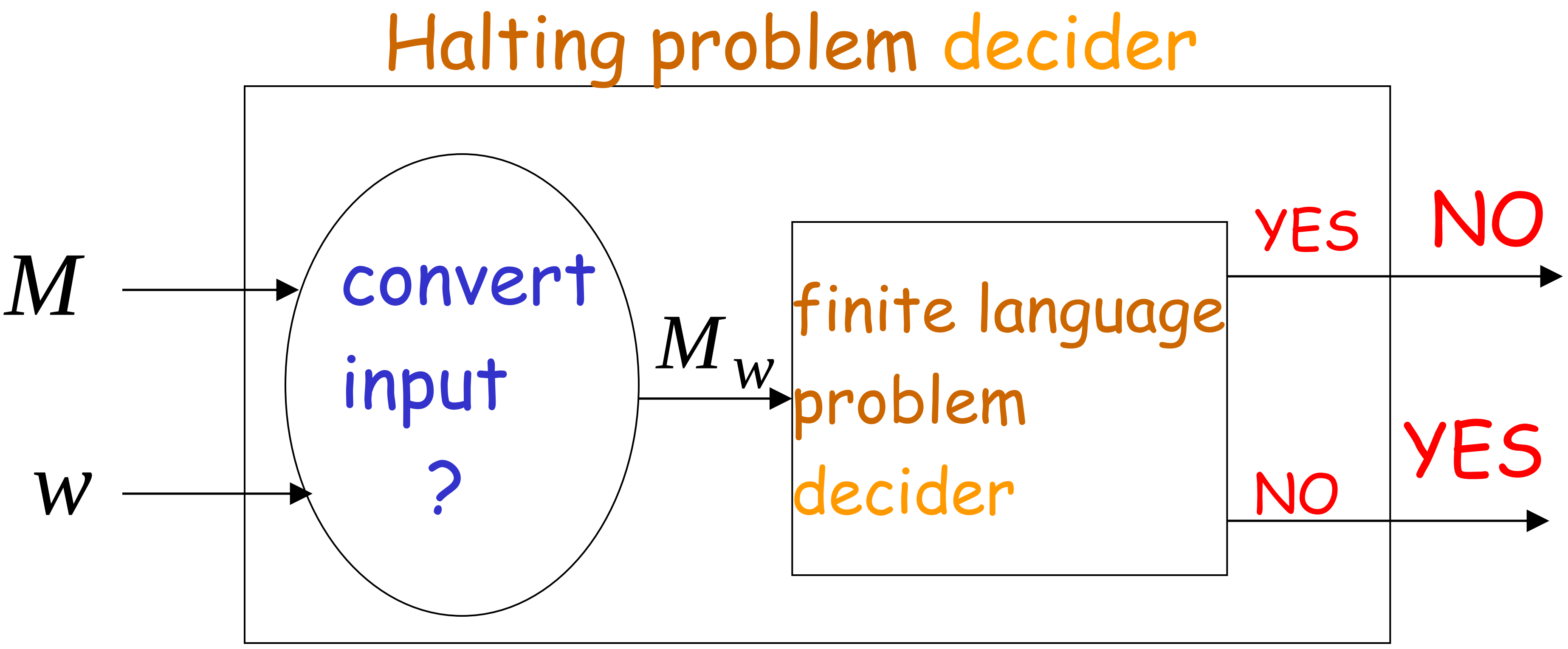
We will build a decider for the halting problem:



We want to reduce the halting problem to the finite language problem



We need to convert one problem instance to the other problem instance



Construct machine $M_w$:

On arbitrary input string $s$

Initially, simulates $M$ on input $w$

If $M$ enters a halt state,
accept $s$ ( $\Sigma^*$ inifinite language)

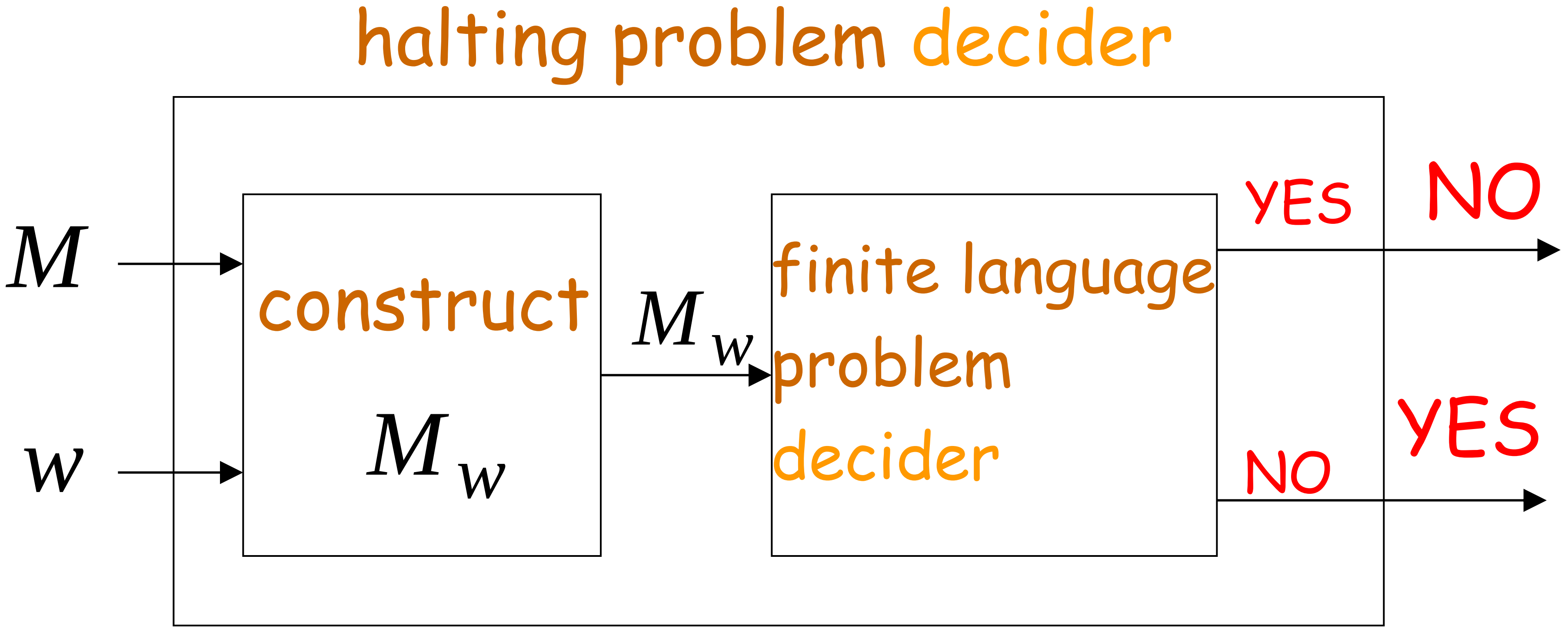
Otherwise, reject $s$ ( $\emptyset$ finite language)

$M$ halts on $w$

if and only if

$L(M_w)$ is infinite

$$L(M_w) = \Sigma^*$$


halting problem decider
M
w
construct
$M_w$
$M_w$
finite language
problem
decider
YES
NO
NO
YES

Take a recursively enumerable language $L$

Decision problems:

- $L$ is empty?
- $L$ is finite?
- $L$ contains two different strings of the same length?

All these problems are undecidable

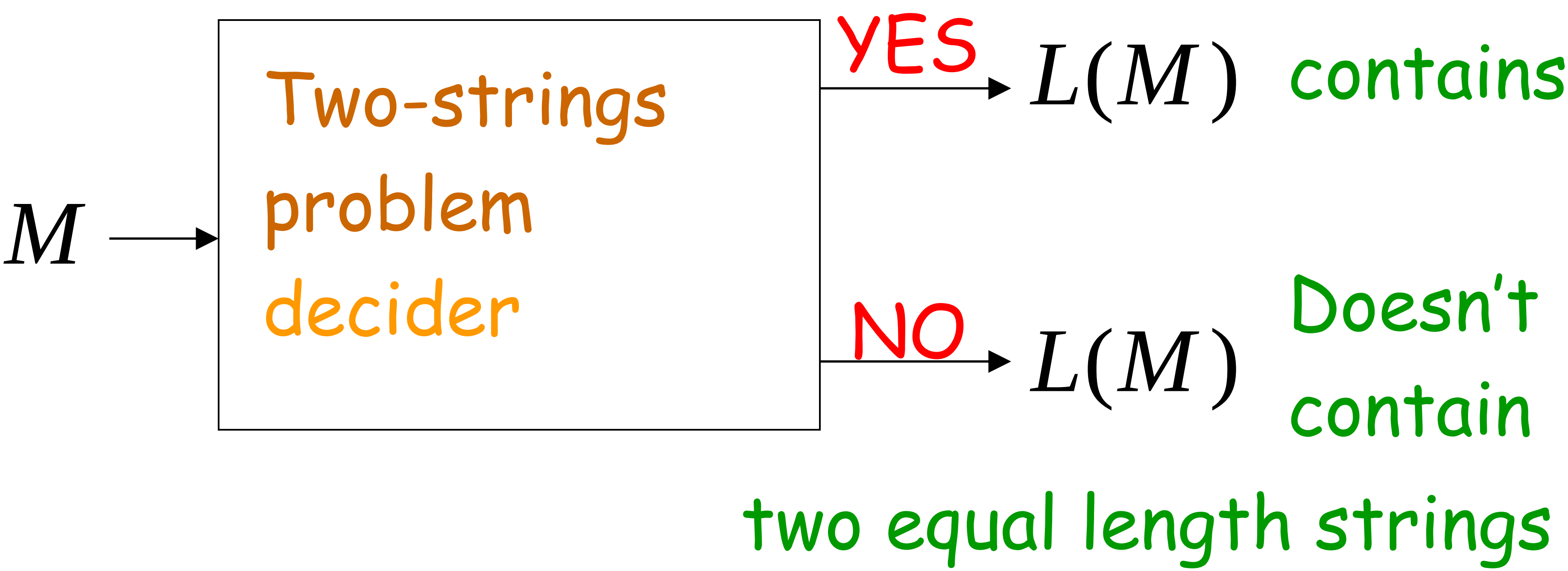
**Theorem:**

For a recursively enumerable language $L$ it is undecidable to determine whether $L$ contains two different strings of same length

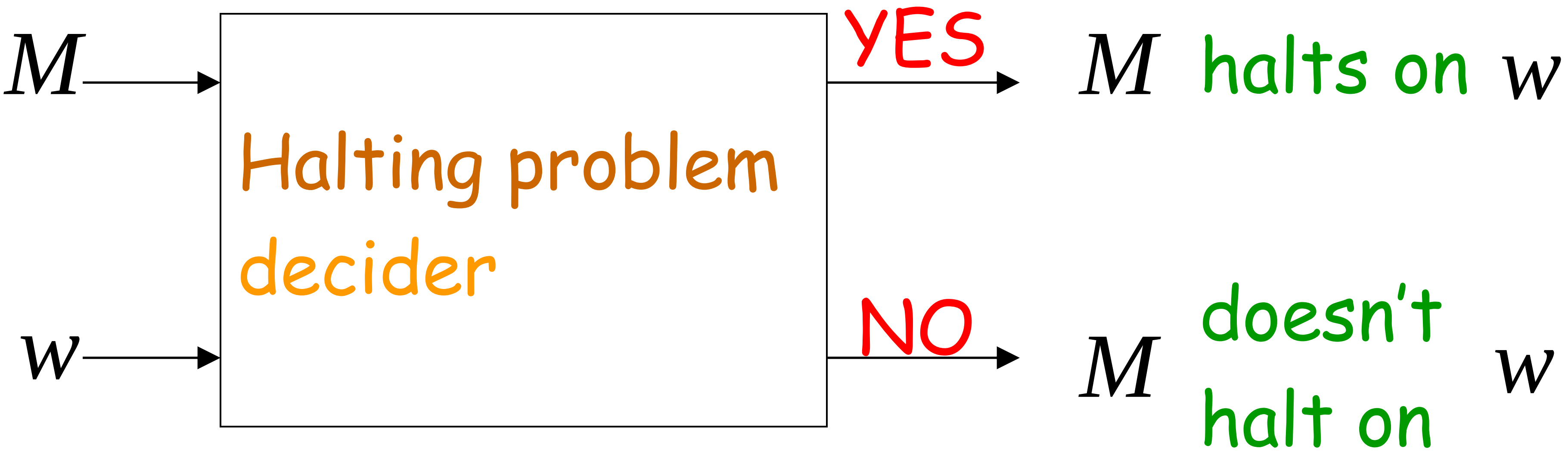
**Proof:** We will reduce the halting problem to this problem

Let $M$ be the TM with $L(M) = L$

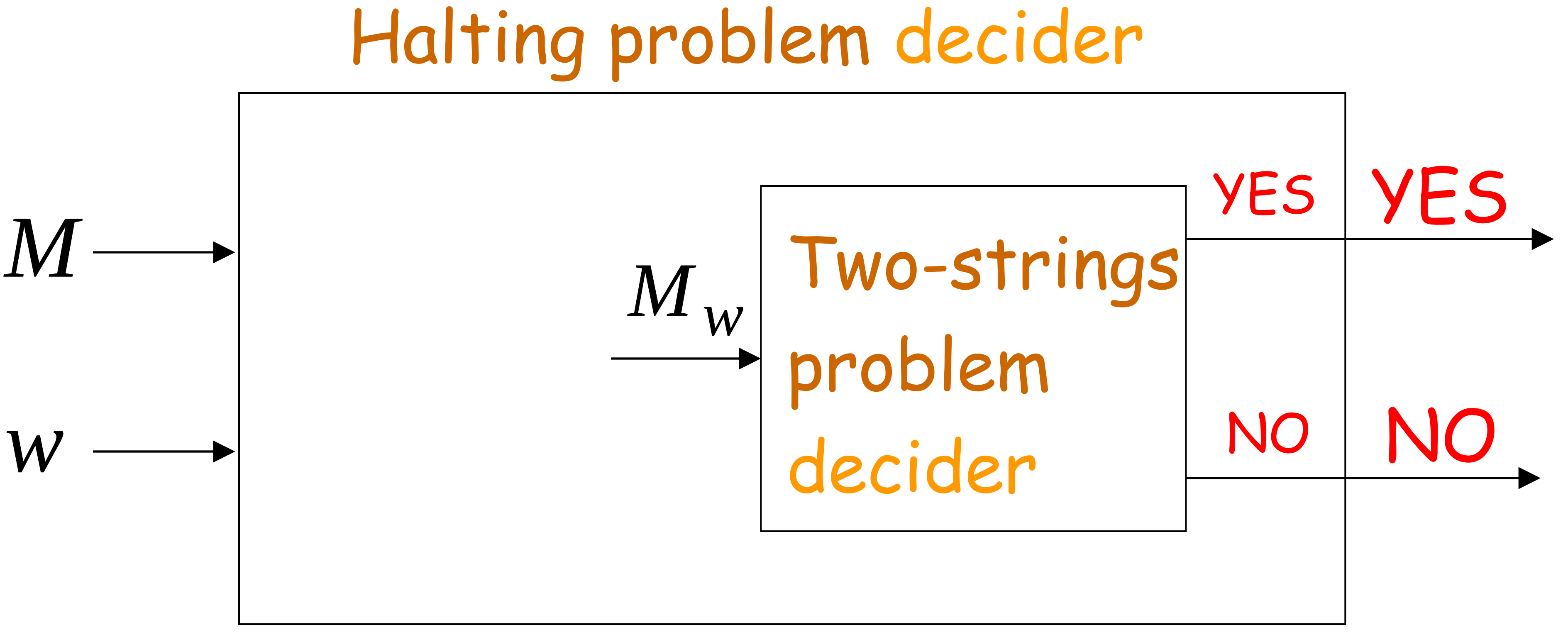
Suppose we have the decider for the two-strings problem:



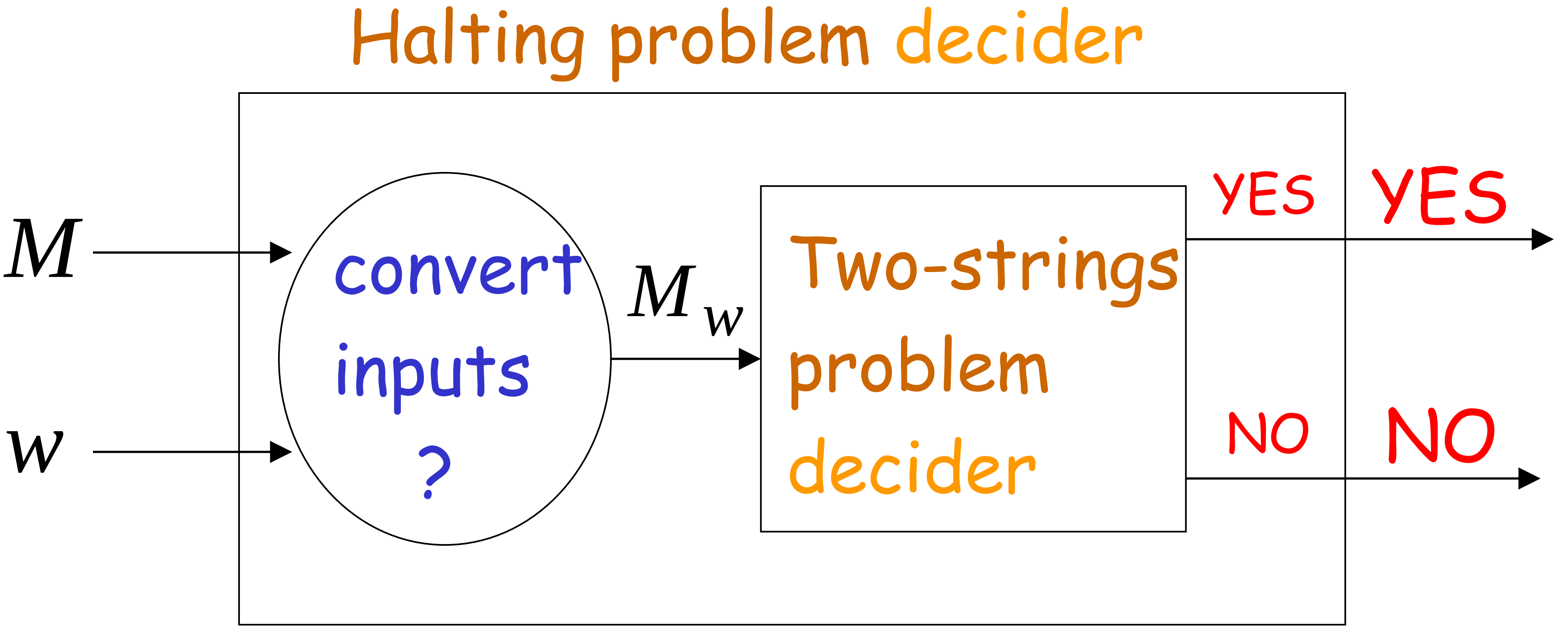
We will build a decider for the halting problem:



We want to reduce the halting problem to the empty language problem



We need to convert one problem instance to the other problem instance



Construct machine $M_w$ :

On arbitrary input string $s$

Initially, simulate $M$ on input $w$

When $M$ enters a halt state,
accept if $s = a$ or $s = b$
(two equal length strings $L(M_w) = \{a, b\}$ )

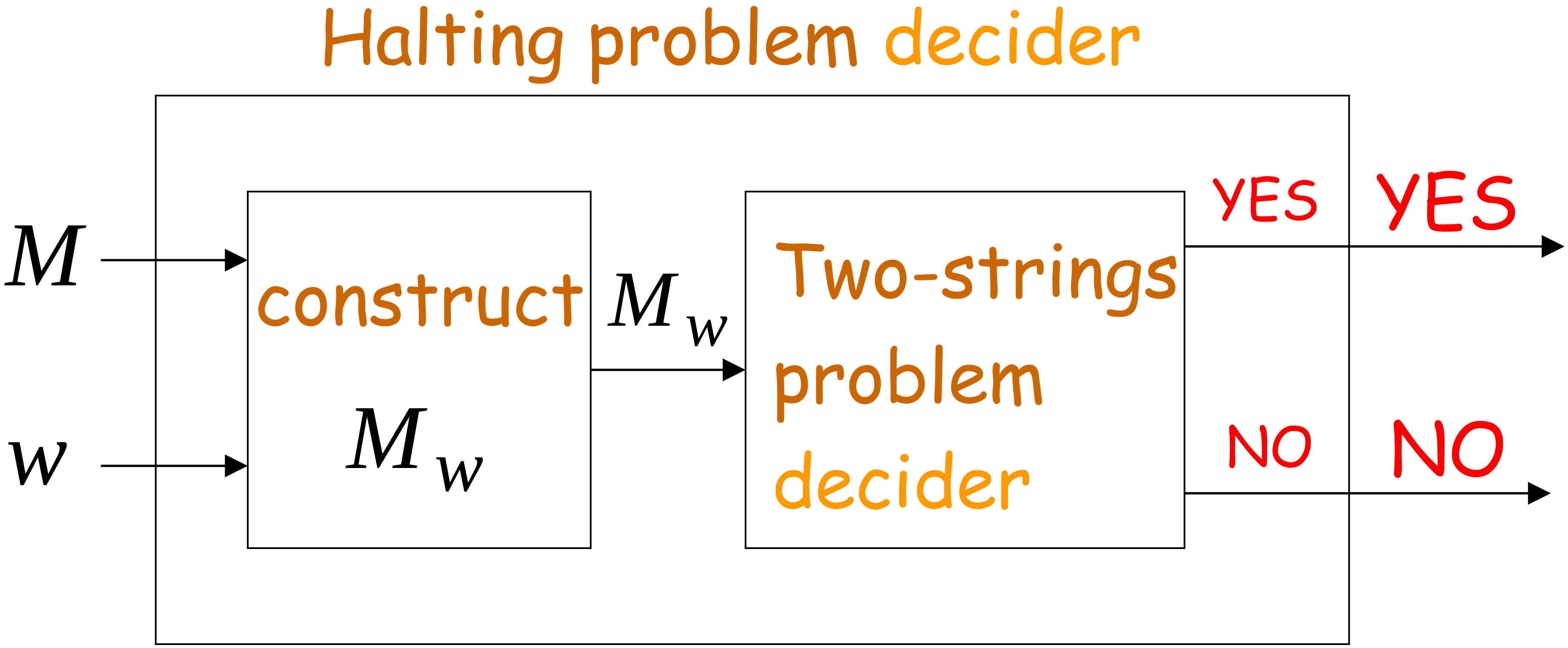
Otherwise, reject $s$ ( $L(M_w) = \emptyset$ )

$M$ halts on $w$

if and only if

$M_w$ accepts two equal length strings

$M_w$ accepts $a$ and $b$


Halting problem decider
$M$
$w$
construct
$M_w$
$M_w$
Two-strings
problem
decider
YES
YES
NO
NO

# Rice's Theorem

## Definition:

Non-trivial properties of recursively enumerable languages:

any property possessed by some (not all) recursively enumerable languages

Some non-trivial properties of recursively enumerable languages:

- $L$ is empty
- $L$ is finite
- $L$ contains two different strings of the same length

# Rice's Theorem:

Any non-trivial property of
a recursively enumerable language
is undecidable

# The Turing Test

- The phrase "The Turing Test" is most properly used to refer to a proposal made by Turing (1950) as a way of dealing with the question whether machines can think.

# Turing test description

- Suppose that we have a person, a machine, and an interrogator.
- The interrogator is in a room separated from the other person and the machine.
- The object of the game is for the interrogator to determine which of the other two is the person, and which is the machine.

# Turing test description (Cont.)

- The interrogator knows the other person and the machine by the labels X and Y
- but, at least at the beginning of the game, does not know which of the other person and the machine is X
- and at the end of the game says either 'X is the person and Y is the machine' or 'X is the machine and Y is the person'.

# Turing test description (Cont.)

- The interrogator is allowed to put questions to the person and the machine of the following kind:
- "Will X please to tell me whether X plays chess?" Whichever of the machine and the other person is X must answer questions that are addressed to X.

# Turing test description (Cont.)

- The object of the machine is to try to cause the interrogator to mistakenly conclude that the machine is the other person;
- the object of the other person is to try to help the interrogator to correctly identify the machine. About this game

# Questions

- There are at least two kinds of questions that can be raised about Turing test.

# Questions (Cont.)

- First, there are empirical questions, e.g., Is it true that we now -- or will soon -- have made computers that can play the imitation game so well that an average interrogator has no more than a 70 percent chance of making the right identification after five minutes of questioning?

# Questions (Cont.)

- Second, there are conceptual questions,
- if an average interrogator had no more than a 70 percent chance of making the right identification after five minutes of questioning,
- we should conclude that the machine exhibits some level of thought, or intelligence, or mentality?
- Turing tought if it’ s true, machines can think!

# Thinking!?

- we should conclude that the machine exhibits some level of thought, or intelligence, or mentality?

- Turing tought if it’ s true, machines can think!

# Chinese Room

- There are many different objections to The Turing Test during the past fifty plus years.
- We cannot hope to canvass all of these objections here.

# Chinese Room (Cont.)

- However, in 1984, John Searle published a book, 《Minds, Brains and Science》;
- There is one argument—John Searle’s “Chinese Room” in this book;
- That is mentioned so often in connection with the Turing Test.

# Chinese Room (Cont.)

- A digital computer simulates intelligence but in which the digital computer does not itself possess intelligence.
- Searle is here disagreeing with Turing's claim that an appropriately programmed computer could think.

# Chinese Room (Cont.)

- The human in the Chinese Room follows English instructions for manipulating Chinese symbols, where a computer "follows" a program written in a computing language.
- Since a computer just does what the human does — manipulate symbols on the basis of their syntax alone - no computer, merely by following a program, comes to genuinely understand Chinese.