Reductions

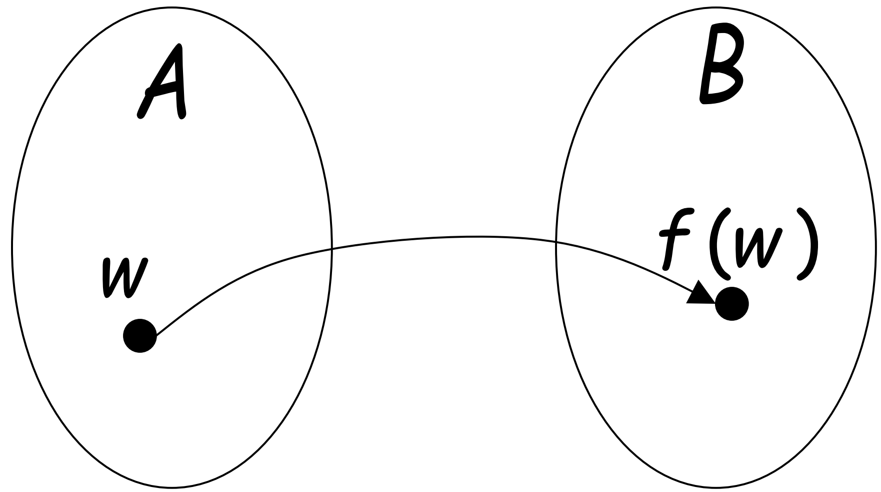
Problem X is reduced to problem Y

If we can solve problem Y

then we can solve problem X

Definition:

Language A
is reduced to
language B



There is a computable
function f (*reduction*) such that:

$$w \in A \iff f(w) \in B$$

Recall:

Computable function f:

There is a deterministic Turing machine M which for any string w computes $f(w)$

Theorem:

<u>If</u>: a: Language A is reduced to B

b: Language B is decidable

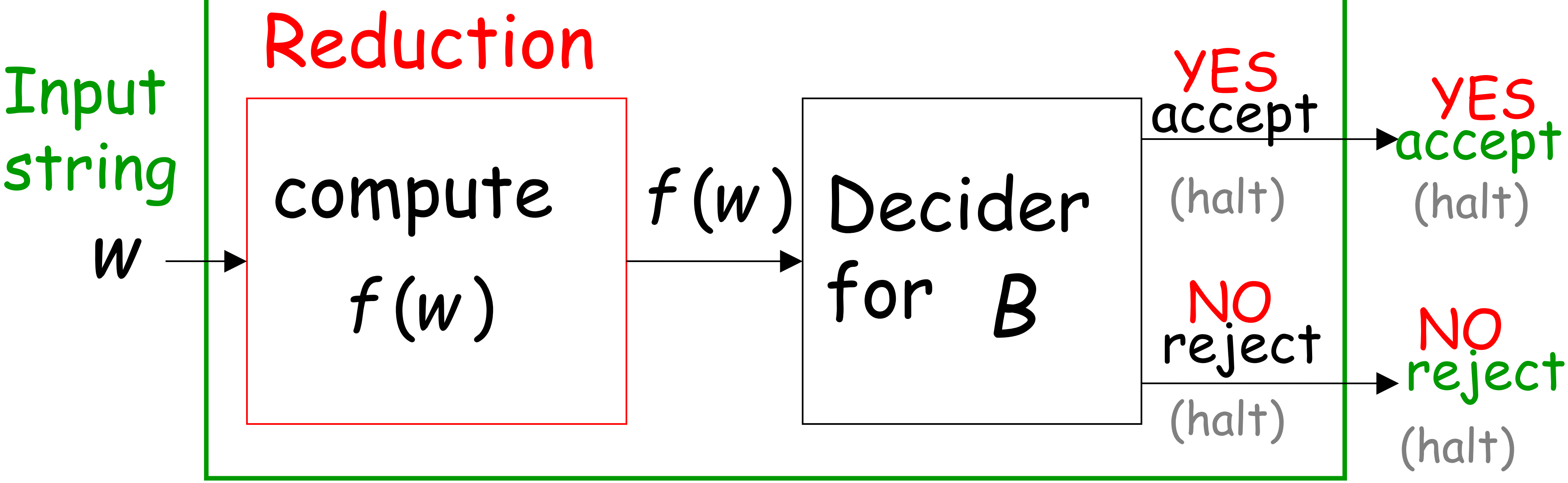
<u>Then</u>: A is decidable

Proof:

Basic idea:

Build the decider for A
using the decider for B

Decider for A



$$w \in A \iff f(w) \in B$$

END OF PROOF

Example:

$$EQUAL_{DFA} = \{\langle M_1, M_2 \rangle : M_1 \text{ and } M_2 \text{ are DFAs that accept the same languages}\}$$

is reduced to:

$$EMPTY_{DFA} = \{\langle M \rangle : M \text{ is a DFA that accepts the empty language } \emptyset\}$$

We only need to construct:

$\langle M_1, M_2 \rangle$ $\longrightarrow$ Turing Machine for reduction f $\longrightarrow$ $f(\langle M_1, M_2 \rangle) = \langle M \rangle$ DFA

$$\langle M_1, M_2 \rangle \in EQUAL_{DFA} \quad \Leftrightarrow \quad \langle M \rangle \in EMPTY_{DFA}$$

Let L_1 be the language of DFA M_1

Let L_2 be the language of DFA M_2

$\langle M_1, M_2 \rangle \longrightarrow$ Turing Machine for reduction f $\longrightarrow f(\langle M_1, M_2 \rangle)$

$= \langle M \rangle$ DFA

construct DFA M

by combining M_1 and M_2 so that:

$$L(M) = (L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2)$$

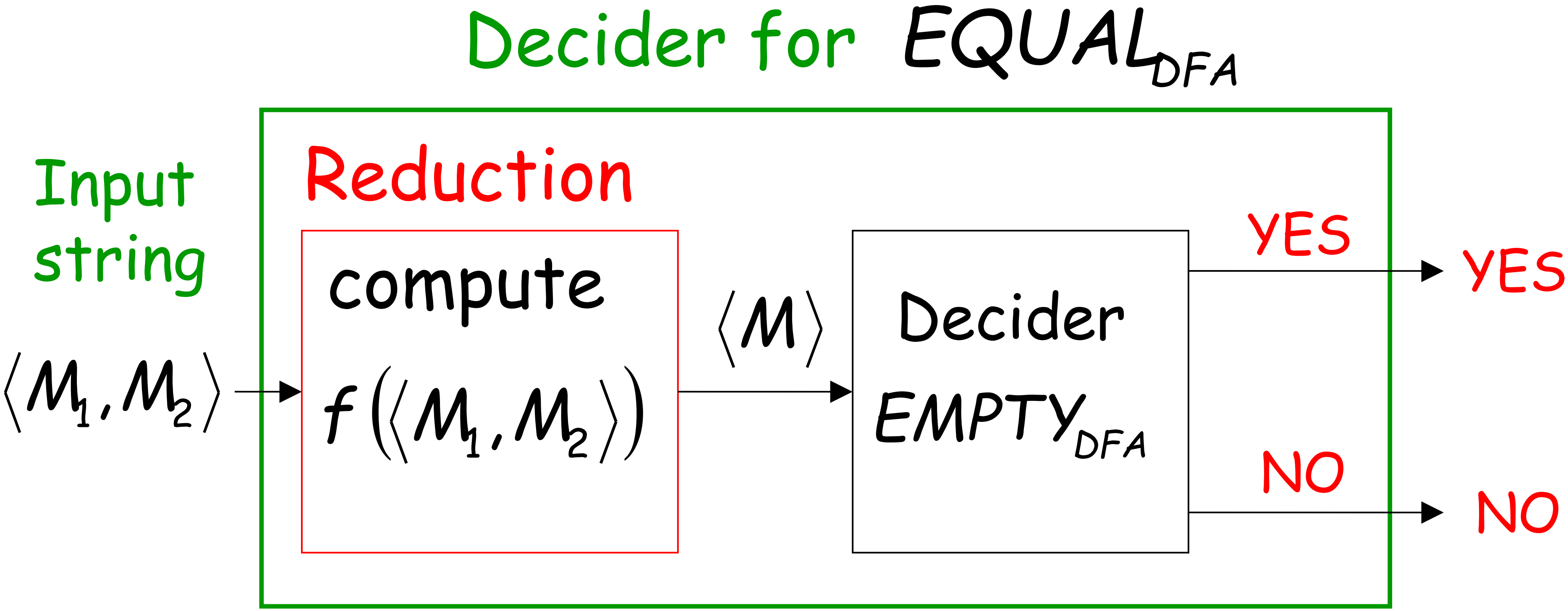
$$L(M) = (L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2)$$

$$\Updownarrow$$

$$L_1 = L_2 \quad \Leftrightarrow \quad L(M) = \emptyset$$

$$\Updownarrow$$

$$\langle M_1, M_2 \rangle \in EQUAL_{DFA} \quad \Leftrightarrow \quad \langle M \rangle \in EMPTY_{DFA}$$



Decider for $EQUAL_{DFA}$
Input string
$\langle M_1, M_2 \rangle$
Reduction
compute
$f(\langle M_1, M_2 \rangle)$
$\langle M \rangle$
Decider
$EMPTY_{DFA}$
YES
YES
NO
NO

Theorem (version 1):

If: a: Language A is reduced to B

b: Language A is undecidable

Then: B is undecidable

(this is the negation of the previous theorem)

Proof: Suppose B is decidable

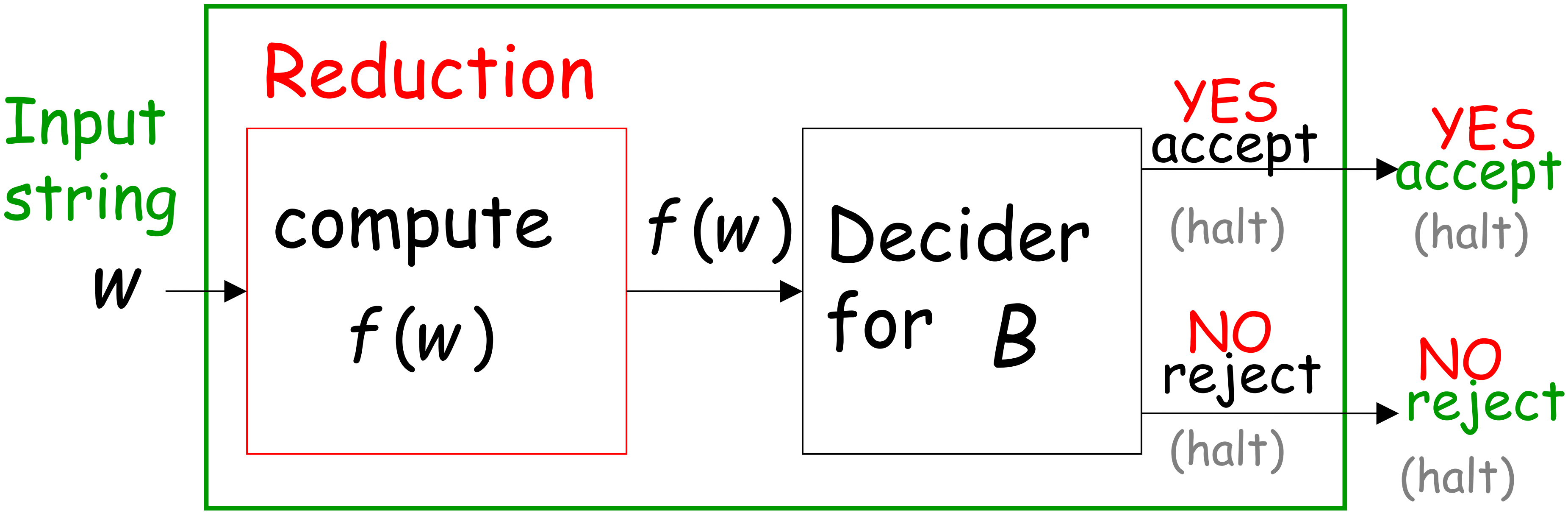
Using the decider for B

build the decider for A

Contradiction!

If B is decidable then we can build:

Decider for A



$$w \in A \iff f(w) \in B$$

CONTRADICTION!

END OF PROOF

Observation:

In order to prove
that some language B is undecidable

we only need to reduce a
known undecidable language A to B

State-entry problem

Input:
- Turing Machine M
- State q
- String w

Question: Does M enter state q while processing input string w ?

Corresponding language:

$$STATE_{TM} = \{\langle M, w, q \rangle : M \text{ is a Turing machine that enters state } q \text{ on input string } w\}$$

Theorem: $STATE_{TM}$ is undecidable

(state-entry problem is unsolvable)

Proof: Reduce

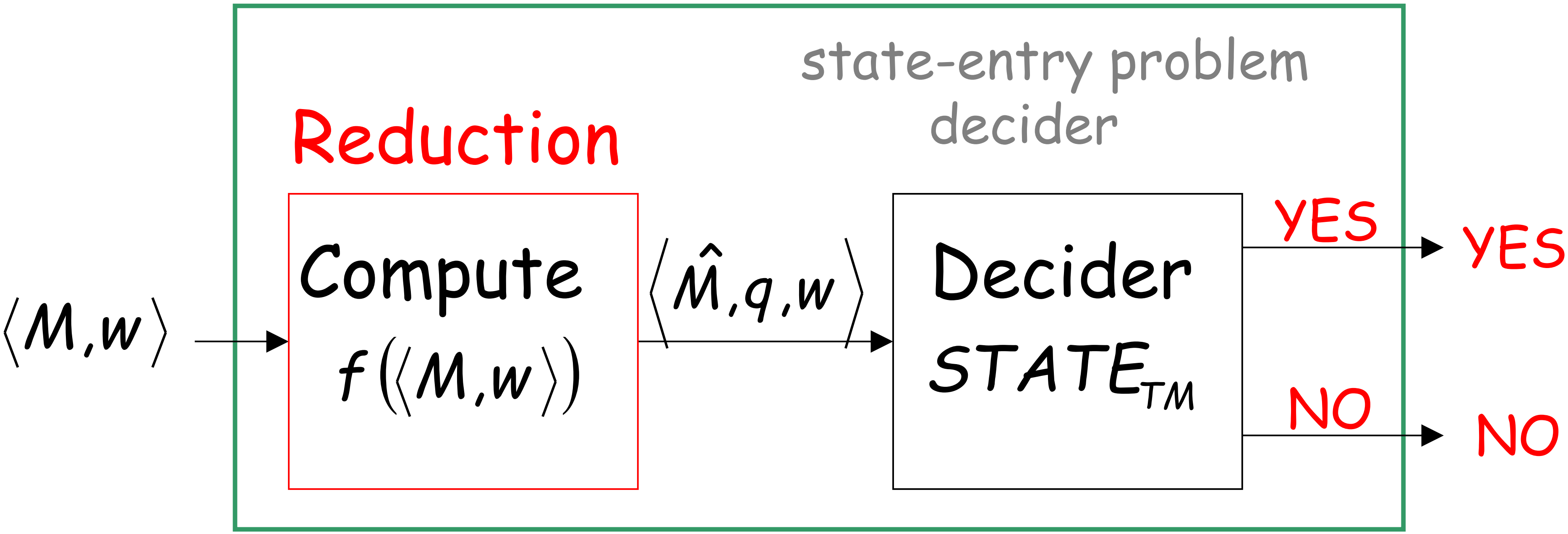
$HALT_{TM}$ (halting problem)

to

$STATE_{TM}$ (state-entry problem)

Halting Problem Decider

Decider for $HALT_{TM}$



Given the reduction,
if $STATE_{TM}$ is decidable,
then $HALT_{TM}$ is decidable

A contradiction!
since $HALT_{TM}$
is undecidable

We only need to build the reduction:

Reduction

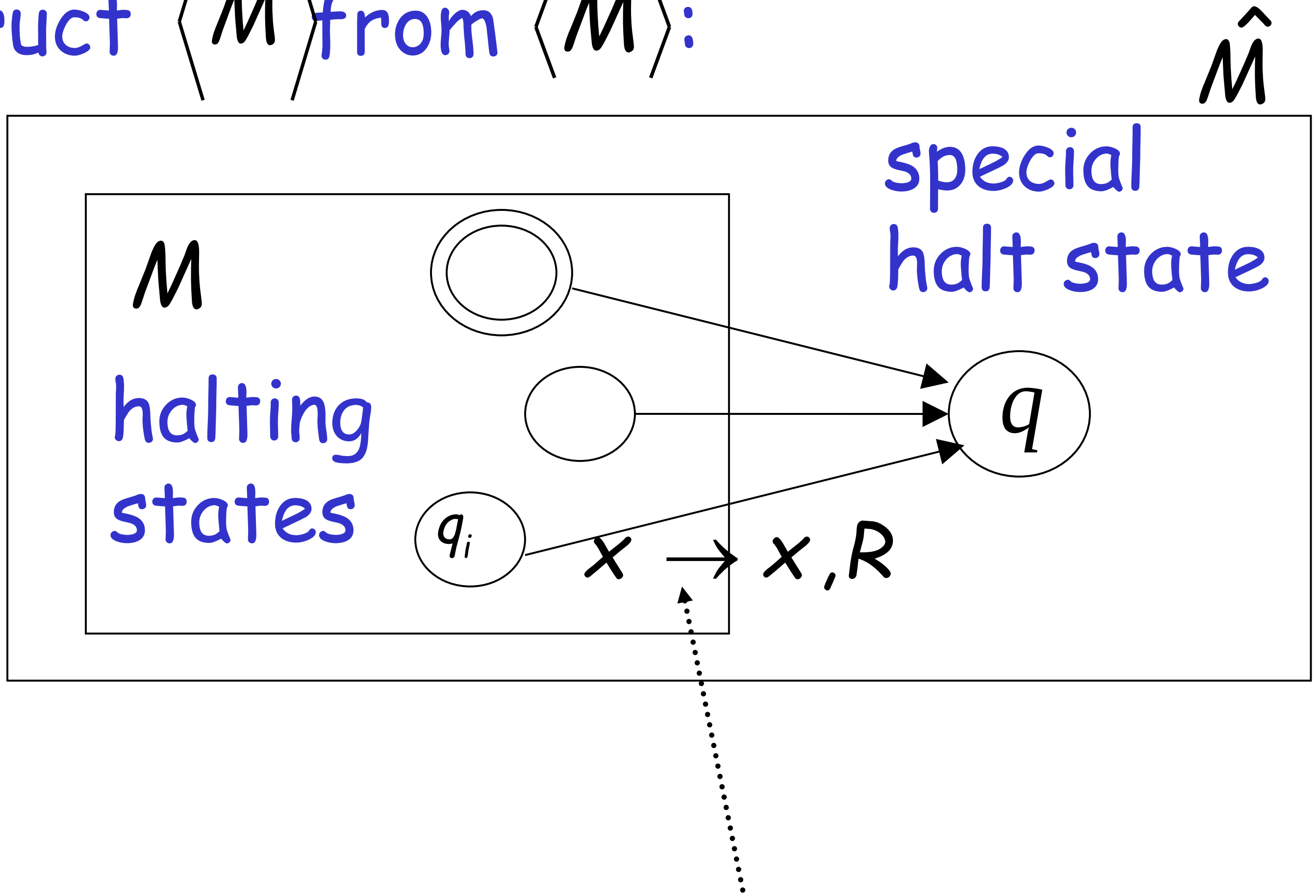
$$\langle M,w \rangle \longrightarrow \boxed{\text{Compute } f(\langle M,w \rangle)} \longrightarrow \langle \hat{M},q,w \rangle$$

$$= f(\langle M,w \rangle)$$

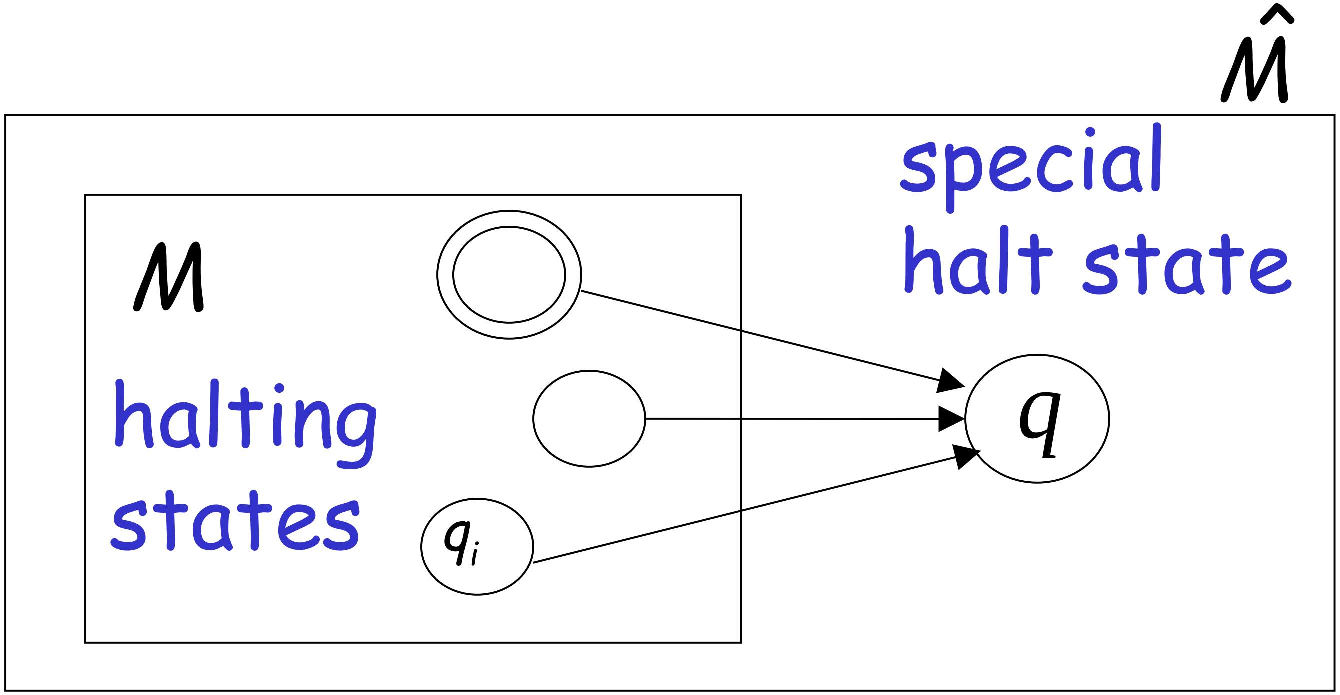
So that:

$$\langle M,w \rangle \in HALT_{TM} \quad \Longleftrightarrow \quad \langle \hat{M},w,q \rangle \in STATE_{TM}$$

Construct $\langle \hat{M} \rangle$ from $\langle M \rangle$:



A transition for every unused tape symbol x of q_i



M halts $\Longleftrightarrow$ $\hat{M}$ halts on state q

Therefore: M halts on input w

$$\Updownarrow$$

$\hat{M}$ halts on state q on input w

Equivalently:

$$\langle M, w \rangle \in HALT_{TM} \iff \langle \hat{M}, w, q \rangle \in STATE_{TM}$$

END OF PROOF

Blank-tape halting problem

Input: Turing Machine M

Question: Does M halt when started with a blank tape?

Corresponding language:

$$BLANK_{TM} = \{\langle M \rangle : M \text{ is a Turing machine that halts when started on blank tape}\}$$

Theorem: $BLANK_{TM}$ is undecidable

(blank-tape halting problem is unsolvable)

Proof: Reduce

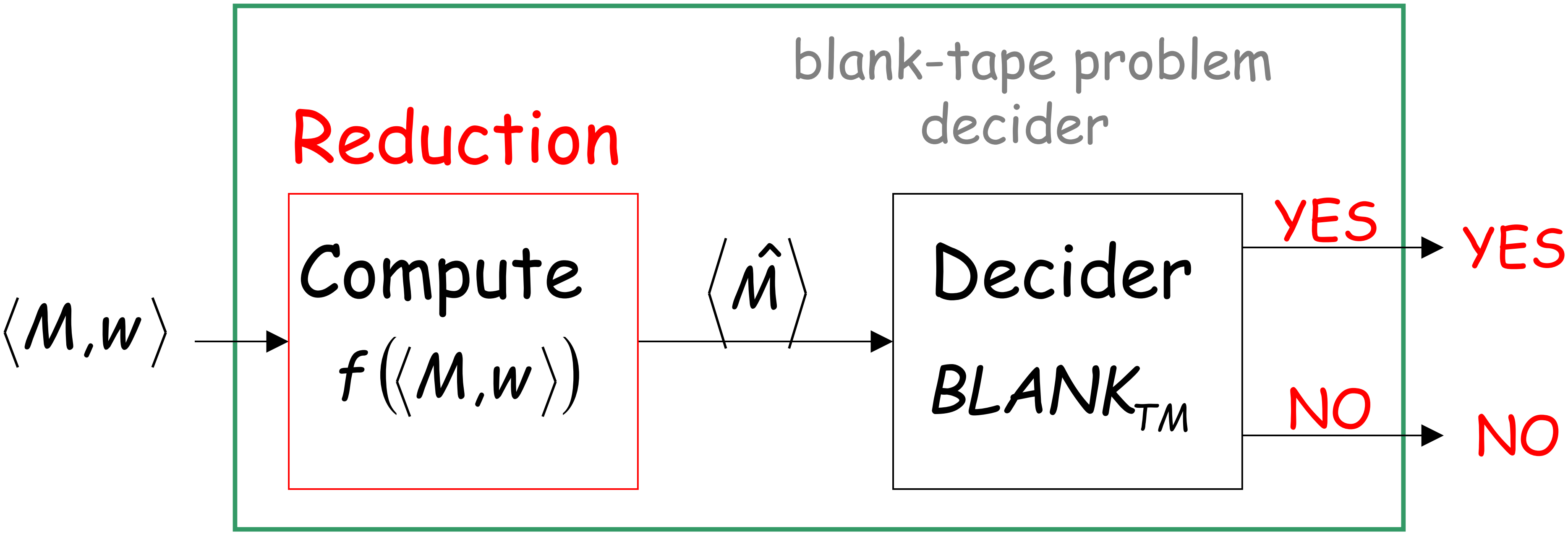
$HALT_{TM}$ (halting problem)

to

$BLANK_{TM}$ (blank-tape problem)

Halting Problem Decider

Decider for $HALT_{TM}$



Given the reduction,
If $BLANK_{TM}$ is decidable,
then $HALT_{TM}$ is decidable

A contradiction!
since $HALT_{TM}$
is undecidable

We only need to build the reduction:

Reduction

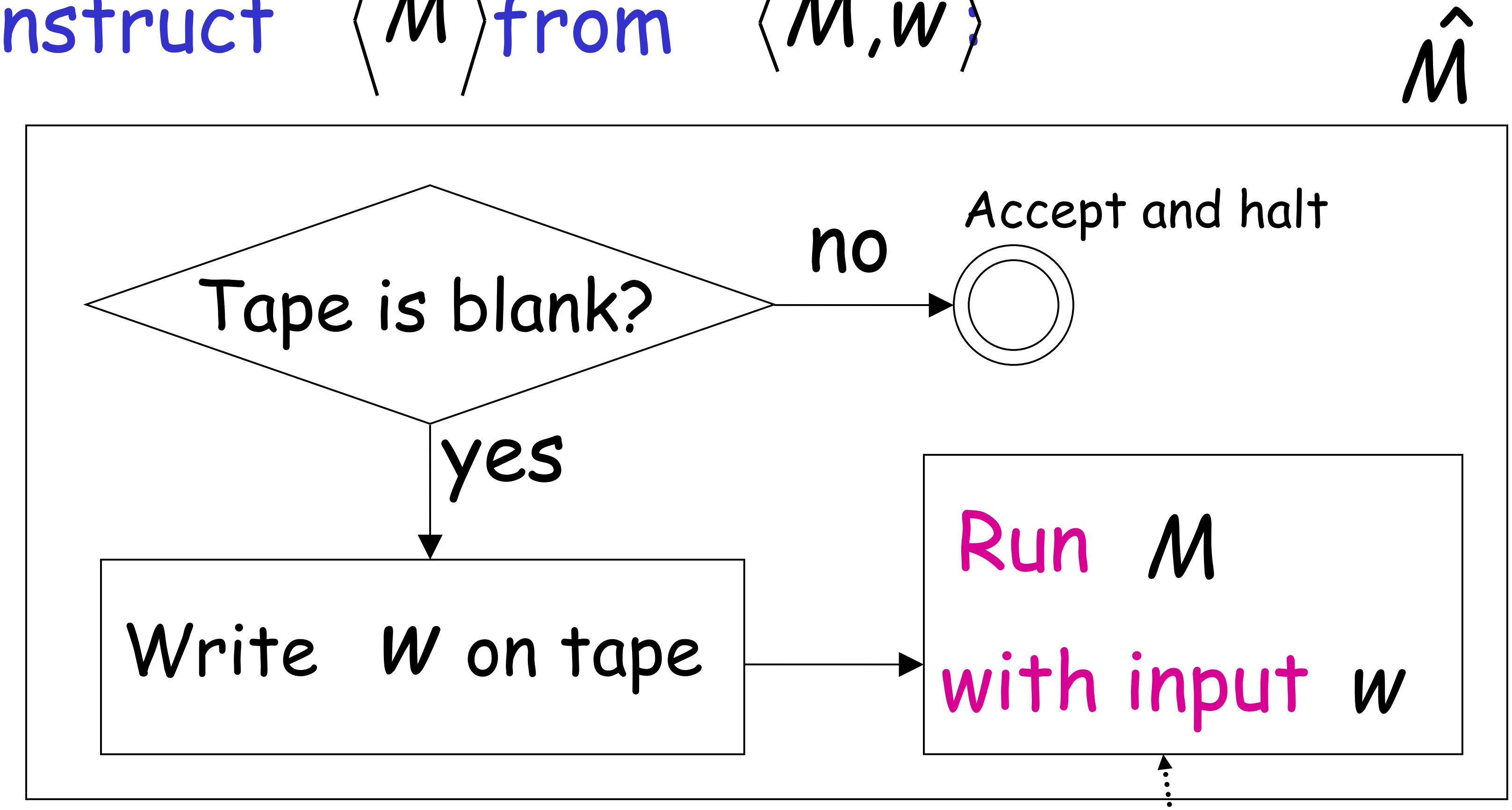
$$\langle M, w \rangle \longrightarrow \boxed{\text{Compute } f(\langle M, w \rangle)} \longrightarrow \langle \hat{M} \rangle$$

$$= f(\langle M, w \rangle)$$

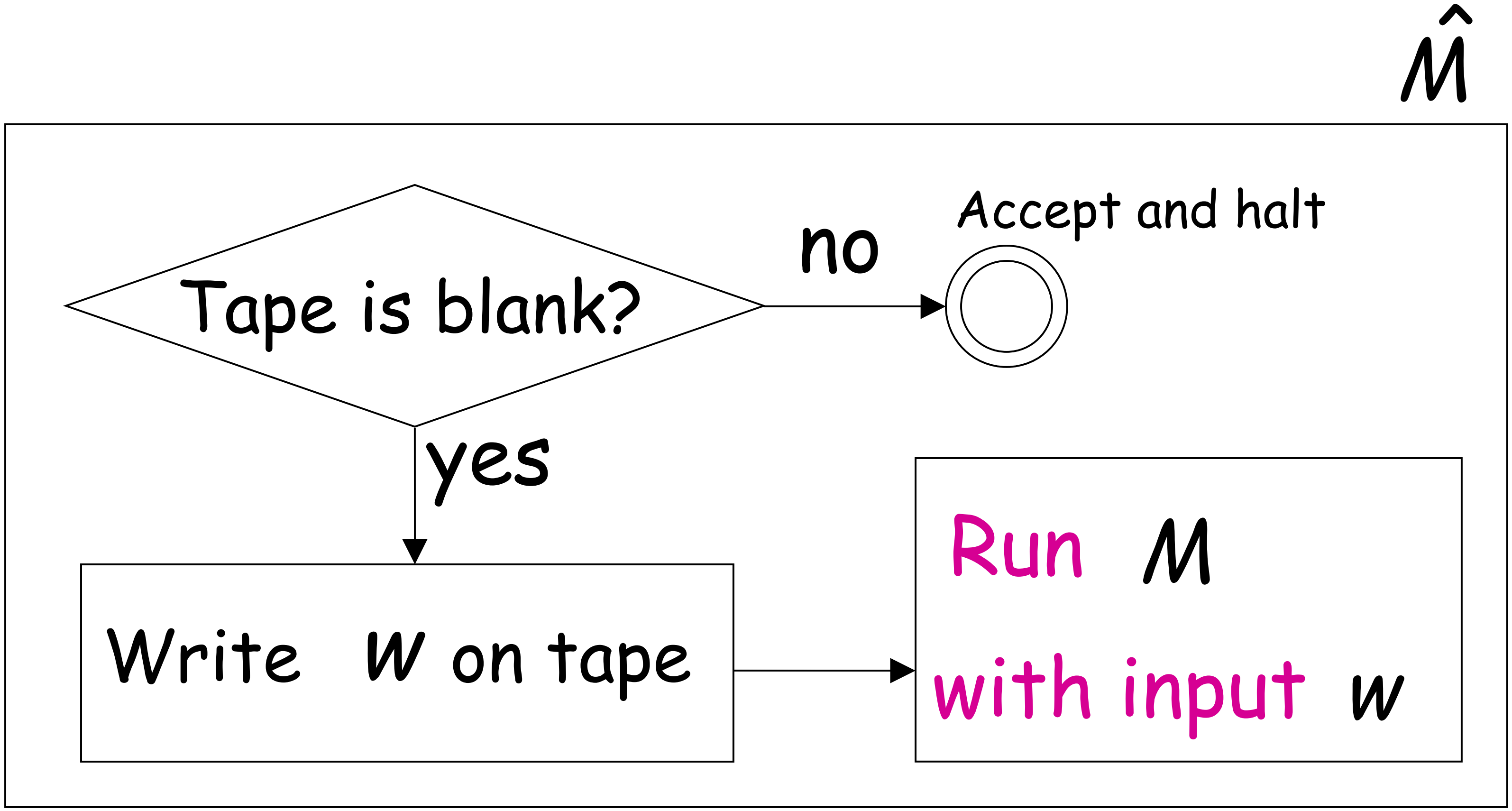
So that:

$$\langle M, w \rangle \in HALT_{TM} \quad \Longleftrightarrow \quad \langle \hat{M} \rangle \in BLANK_{TM}$$

Construct $\langle \hat{M} \rangle$ from $\langle M, w \rangle$:



If M halts then halt



M halts on input w

$\Updownarrow$

$\hat{M}$ halts when started on blank tape

M halts on input w

$\Updownarrow$

$\hat{M}$ halts when started on blank tape

Equivalently:

$$\langle M, w \rangle \in HALT_{TM} \iff \langle \hat{M} \rangle \in BLANK_{TM}$$

END OF PROOF

Theorem (version 2):

If: a: Language A is reduced to $\overline{B}$

b: Language A is undecidable

Then: B is undecidable

Proof: Suppose B is decidable

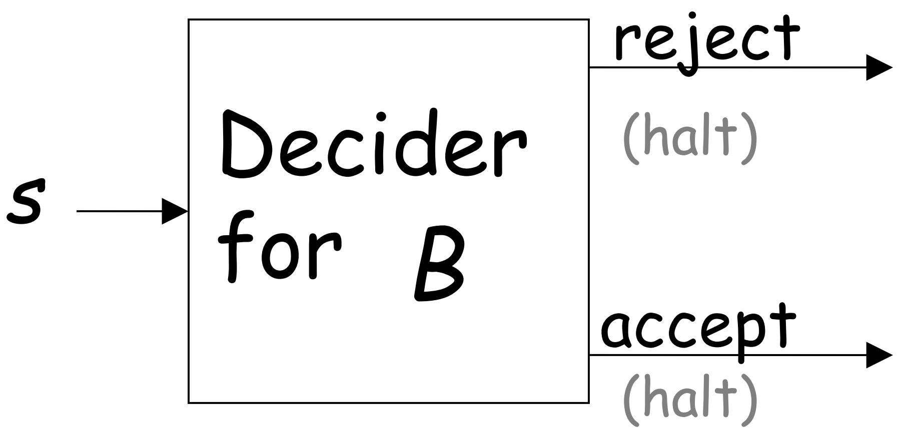
Then $\overline{B}$ is decidable

Using the decider for $\overline{B}$

build the decider for A

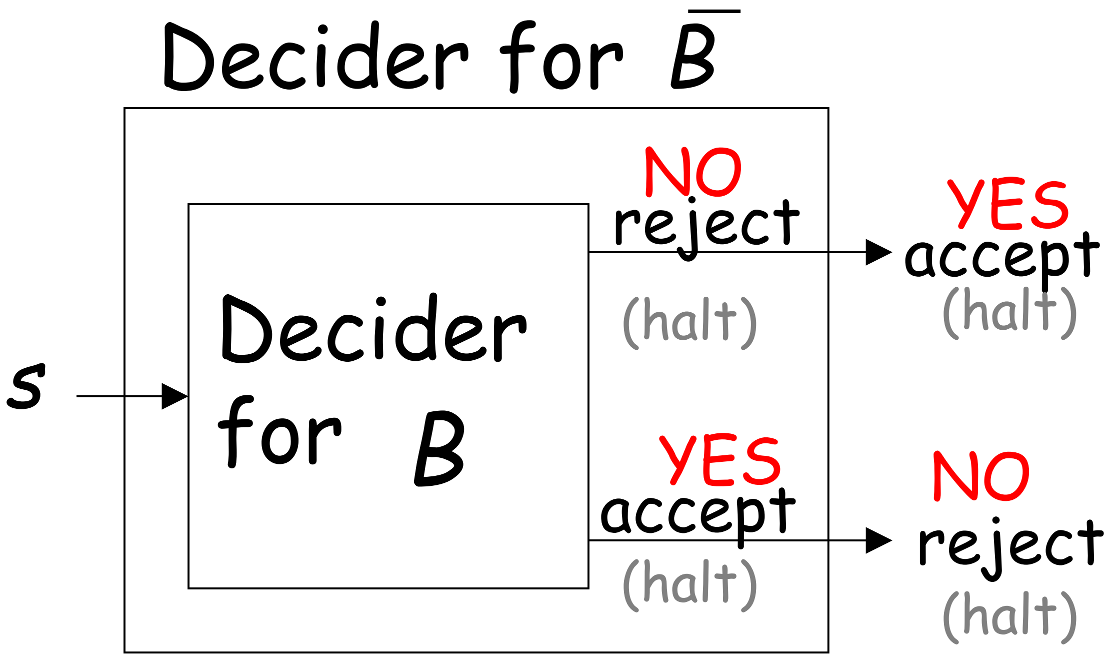
Contradiction!

Suppose B is decidable

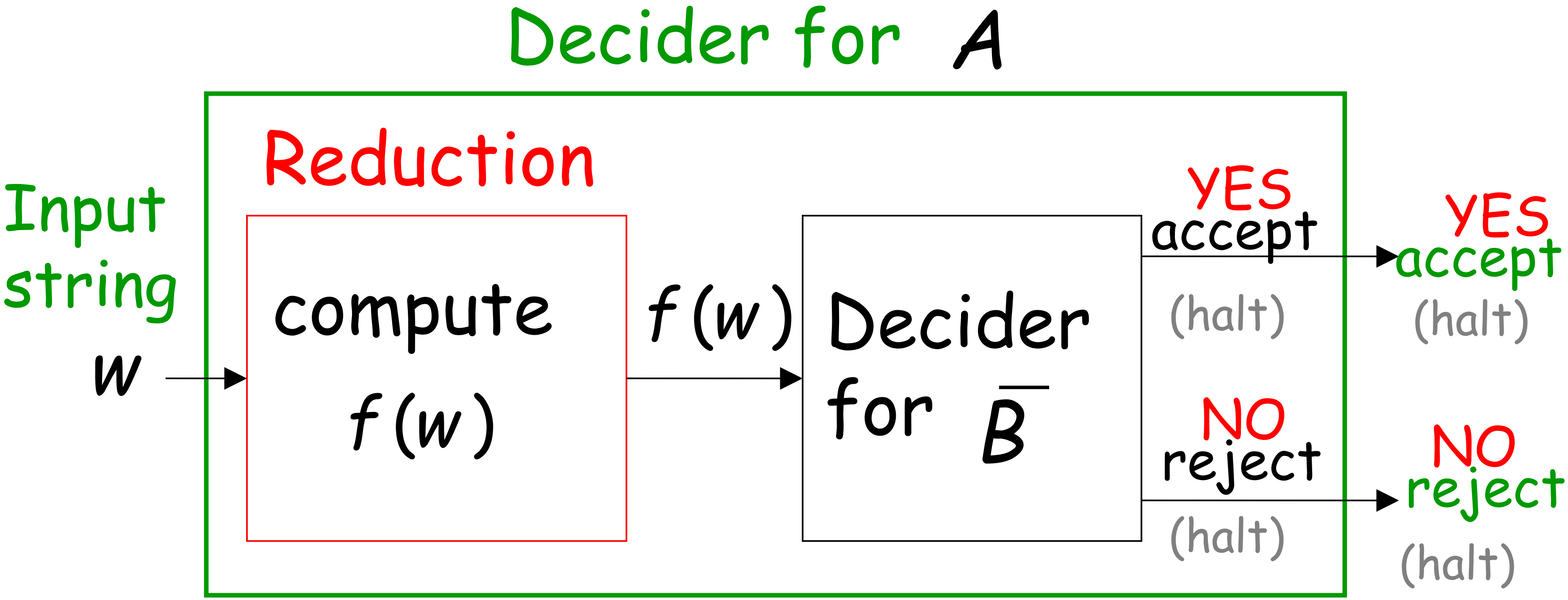


Suppose B is decidable

Then $\overline{B}$ is decidable



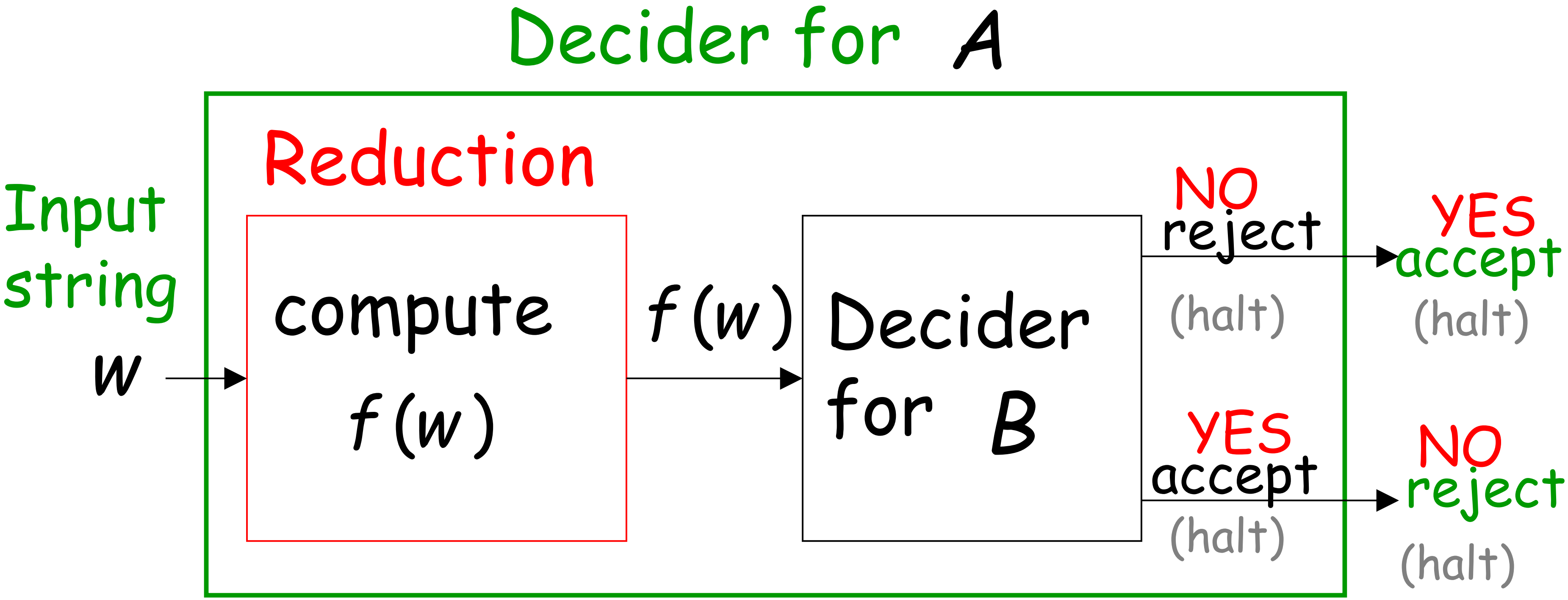
If $\overline{B}$ is decidable then we can build:



$$w \in A \Leftrightarrow f(w) \in \overline{B}$$

CONTRADICTION!

Alternatively:



$$w \in A \Leftrightarrow f(w) \notin B$$

CONTRADICTION!

END OF PROOF

Observation:

In order to prove
that some language B is undecidable
we only need to reduce some
known undecidable language A
to B (theorem version 1)
or to $\overline{B}$ (theorem version 2)

Undecidable Problems for Turing Recognizable languages

Let L be a Turing-acceptable language

- L is empty?
- L is regular?
- L has size 2?

All these are undecidable problems

Let L be a Turing-acceptable language

- L is empty?

- L is regular?

- L has size 2?

Empty language problem

Input: Turing Machine M

Question: Is $L(M)$ empty? $L(M) = \emptyset$?

Corresponding language:

$$EMPTY_{TM} = \{\langle M \rangle : M \text{ is a Turing machine that accepts the empty language } \emptyset\}$$

Theorem: $EMPTY_{TM}$ is undecidable

(empty-language problem is unsolvable)

Proof: Reduce

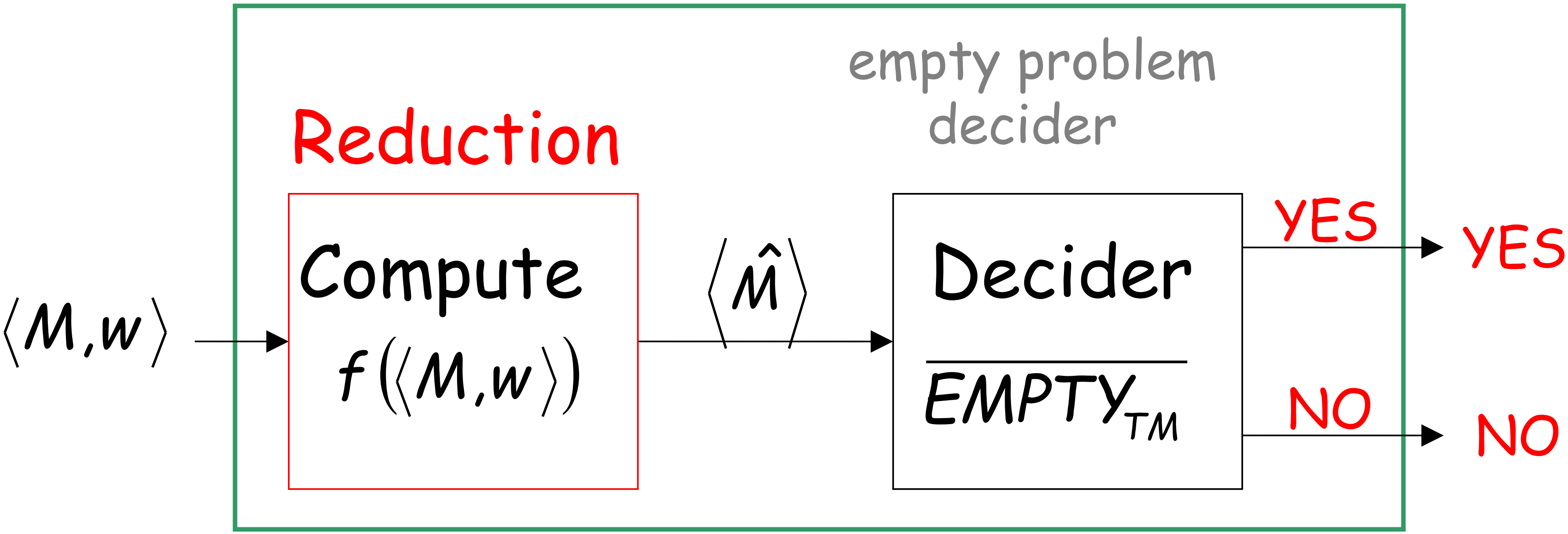
A_{TM} (membership problem)

to

$\overline{EMPTY_{TM}}$ (empty language problem)

membership problem decider

Decider for A_{TM}



Given the reduction,
if $\overline{EMPTY_{TM}}$ is decidable,
then A_{TM} is decidable

A contradiction!
since A_{TM}
is undecidable

We only need to build the reduction:

Reduction

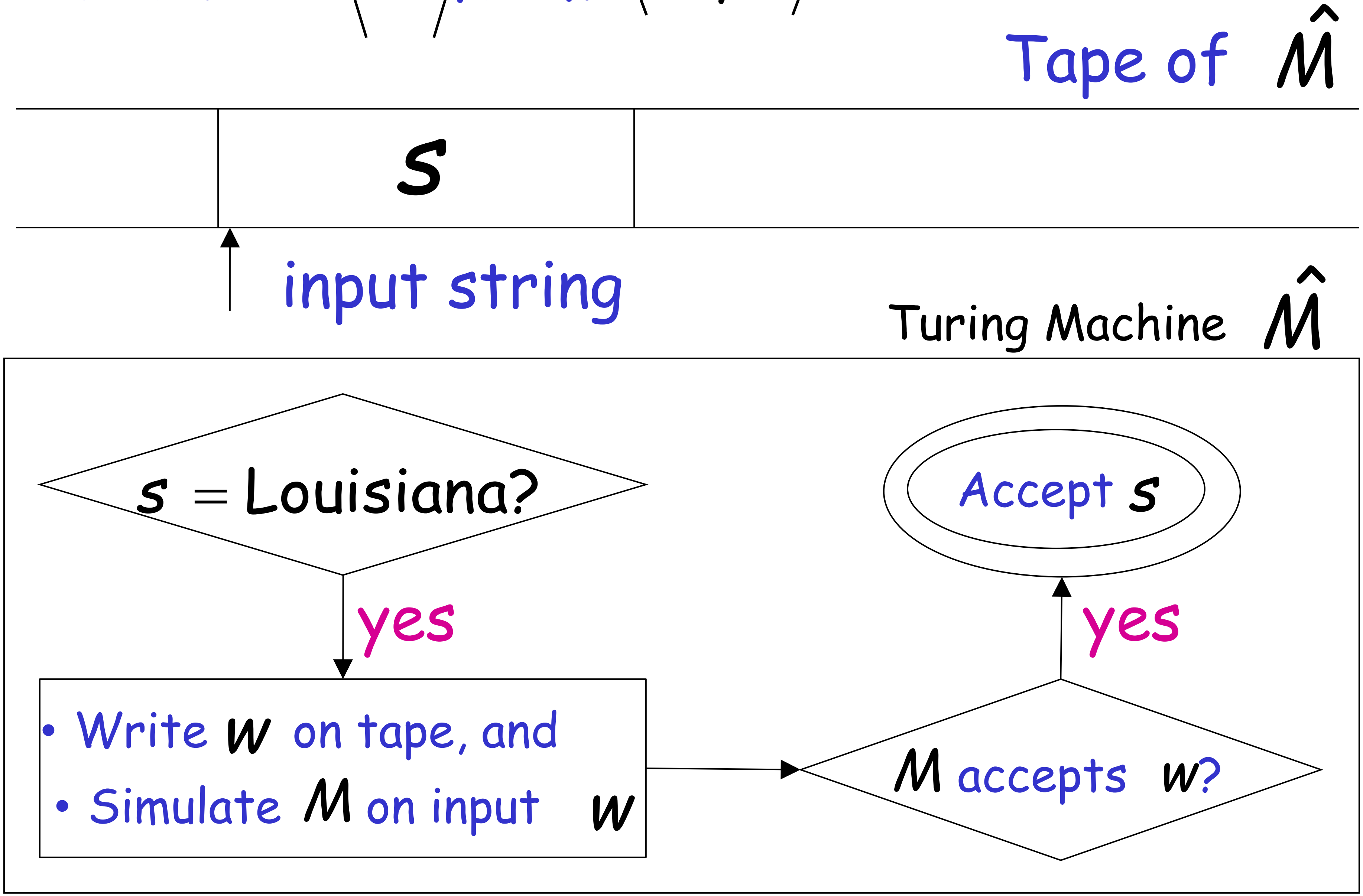
$$\langle M, w \rangle \longrightarrow \boxed{\begin{array}{c}\text{Compute} \\ f(\langle M, w \rangle)\end{array}} \longrightarrow \langle \hat{M} \rangle$$

$$= f(\langle M, w \rangle)$$

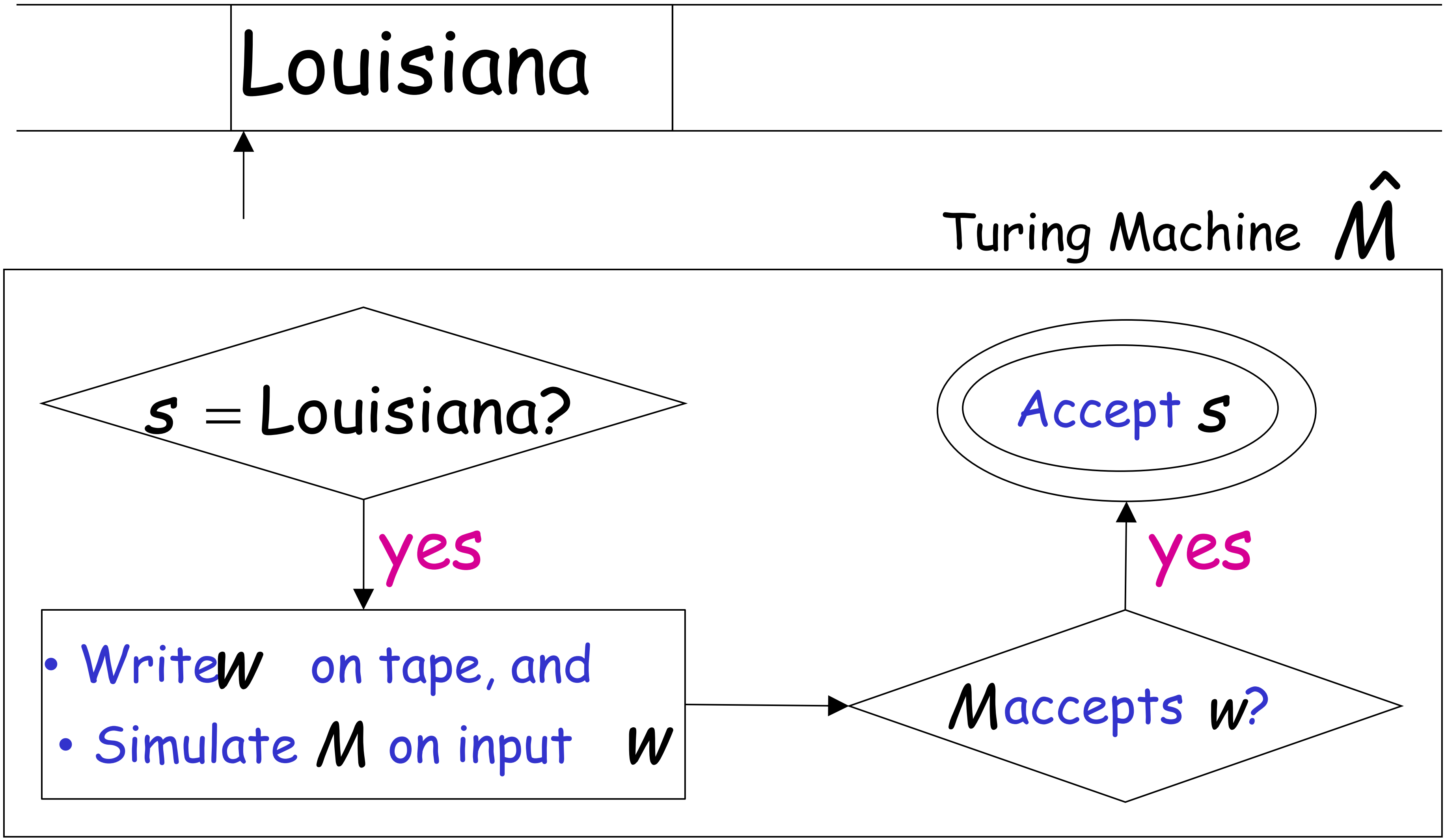
So that:

$$\langle M, w \rangle \in A_{TM} \quad \Longleftrightarrow \quad \langle \hat{M} \rangle \in \overline{EMPTY_{TM}}$$

Construct $\langle \hat{M} \rangle$ from $\langle M, w \rangle$:



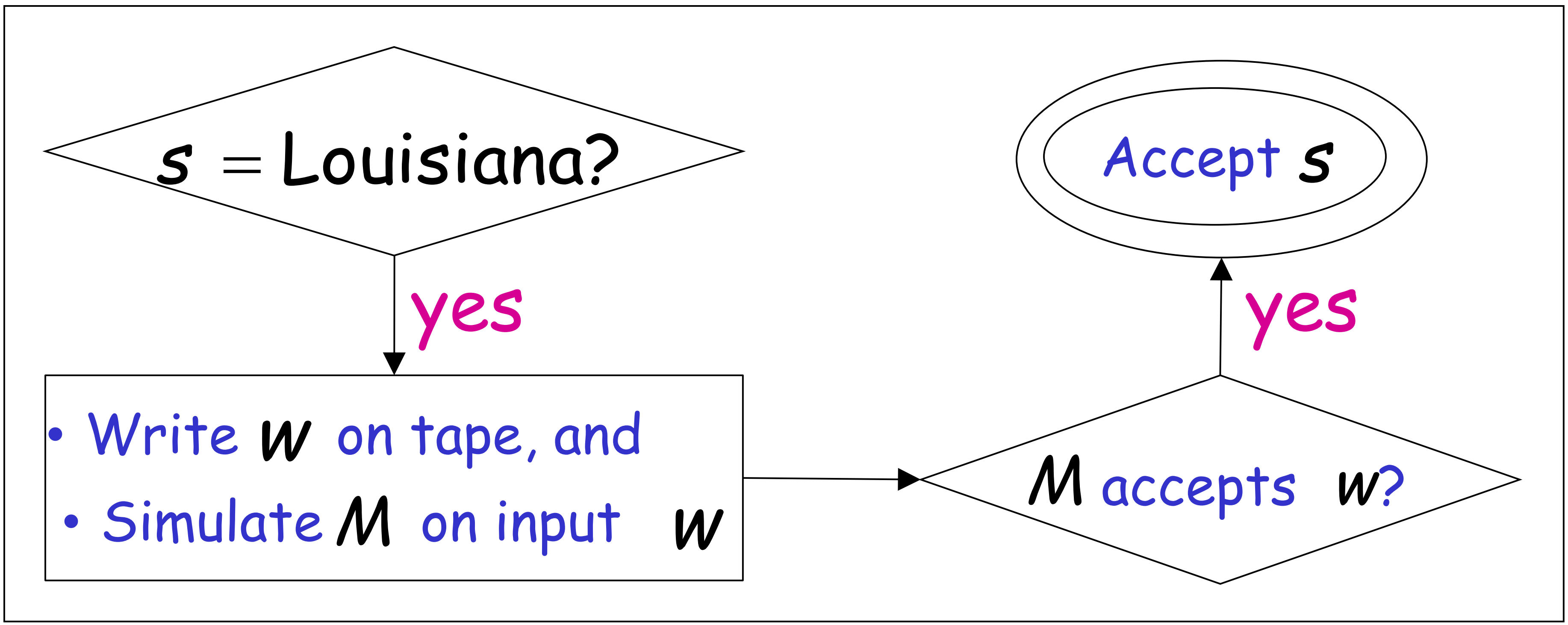
The only possible accepted string s



M accepts w $\Rightarrow$ $L(\hat{M}) = \{\text{Louisiana}\} \neq \emptyset$

M does not accept w $\Rightarrow$ $L(\hat{M}) = \emptyset$

Turing Machine $\hat{M}$



Therefore:

$$M \text{ accepts } w \iff L(\hat{M}) \neq \emptyset$$

Equivalently:

$$\langle M, w \rangle \in A_{TM} \iff \langle \hat{M} \rangle \in \overline{EMPTY_{TM}}$$

END OF PROOF

Let L be a Turing-acceptable language

- L is empty?
- L is regular?
- L has size 2?

Regular language problem

Input: Turing Machine M

Question: Is $L(M)$ a regular language?

Corresponding language:

$$REGULAR_{TM} = \{\langle M \rangle : M \text{ is a Turing machine that accepts a regular language}\}$$

Theorem: $REGULAR_{TM}$ is undecidable

(regular language problem is unsolvable)

Proof: Reduce

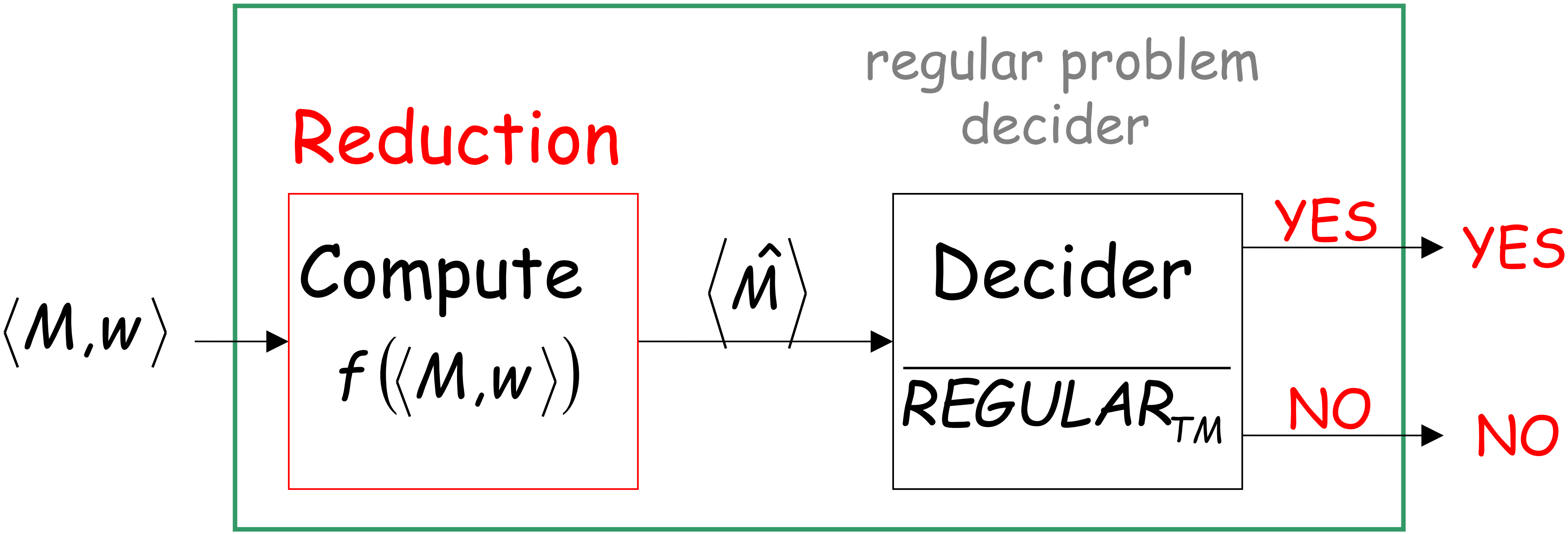
A_{TM} (membership problem)

to

$\overline{REGULAR_{TM}}$ (regular language problem)

membership problem decider

Decider for A_{TM}



Given the reduction,
If $\overline{REGULAR_{TM}}$ is decidable,
then A_{TM} is decidable

A contradiction!
since A_{TM}
is undecidable

We only need to build the reduction:

Reduction

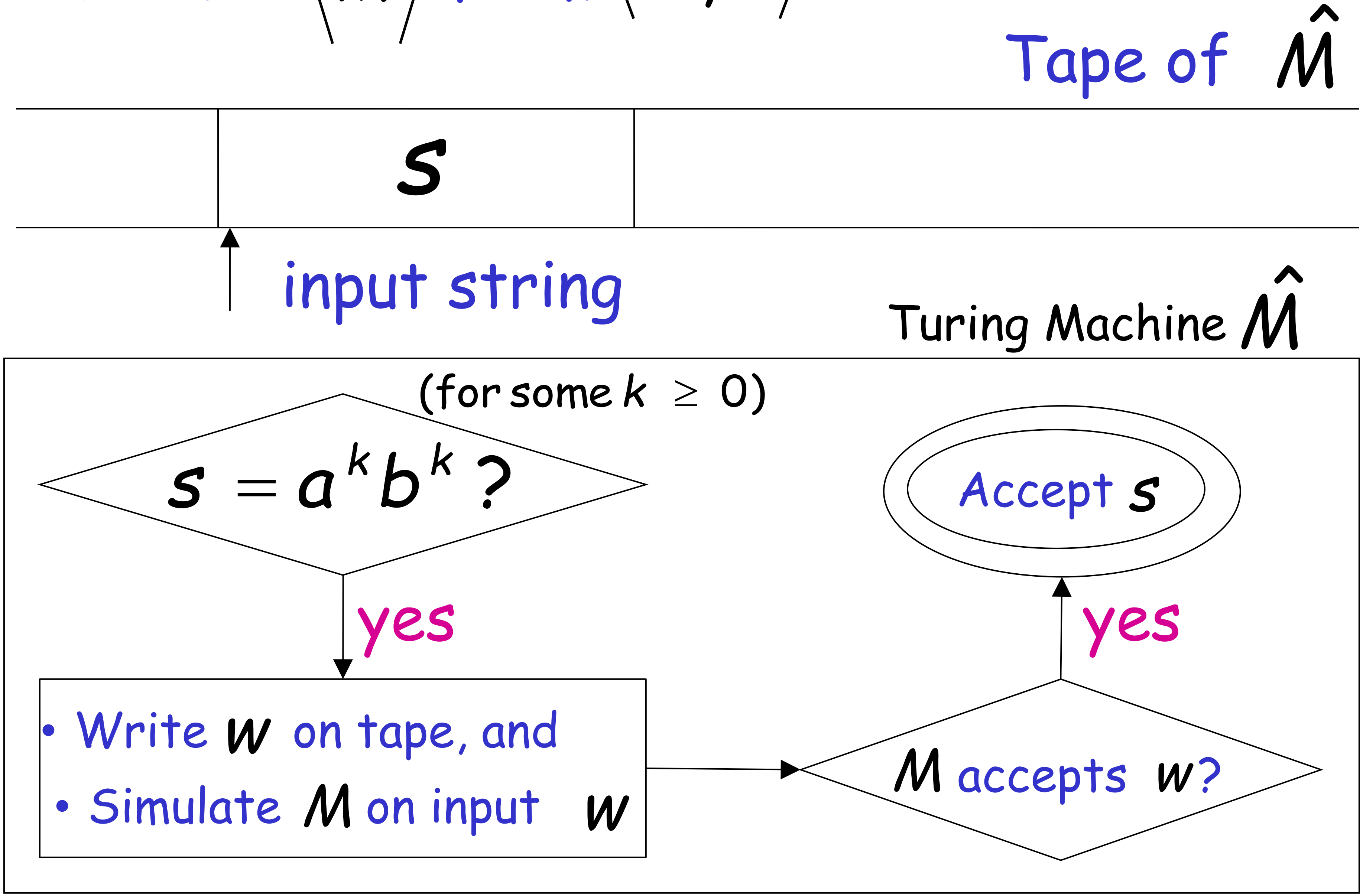
$$\langle M,w \rangle \longrightarrow \boxed{\text{Compute } f(\langle M,w \rangle)} \longrightarrow \langle \hat{M} \rangle$$

$$= f(\langle M,w \rangle)$$

So that:

$$\langle M,w \rangle \in A_{TM} \iff \langle \hat{M} \rangle \in \overline{REGULAR_{TM}}$$

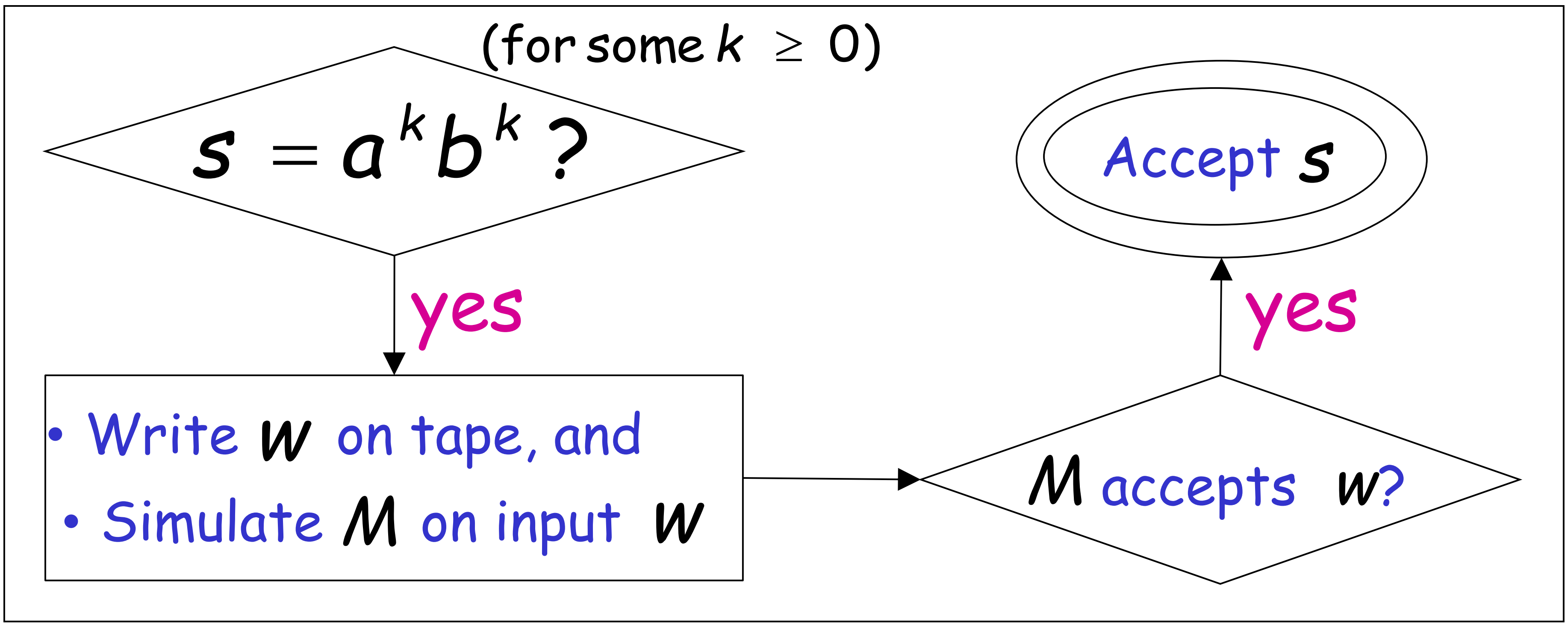
Construct $\langle \hat{M} \rangle$ from $\langle M, w \rangle$:



M accepts w $\Rightarrow$ $L(\hat{M}) = \{a^n b^n : n \geq 0\}$ not regular

M does not accept w $\Rightarrow$ $L(\hat{M}) = \varnothing$ regular

Turing Machine $\hat{M}$



Therefore:

M accepts w $\Longleftrightarrow$ $L(\hat{M})$ is not regular

Equivalently:

$$\langle M, w \rangle \in A_{TM} \Longleftrightarrow \langle \hat{M} \rangle \in \overline{REGULAR_{TM}}$$

END OF PROOF

Let L be a Turing-acceptable language

- L is empty?
- L is regular?
- L has size 2?

Size2 language problem

Input: Turing Machine M

Question: Does $L(M)$ have size 2 (two strings)?

$$|L(M)| = 2?$$

Corresponding language:

$$SIZE2_{TM} = \{\langle M \rangle : M \text{ is a Turing machine that accepts exactly two strings}\}$$

Theorem: $SIZE2_{TM}$ is undecidable

(size2 language problem is unsolvable)

Proof: Reduce

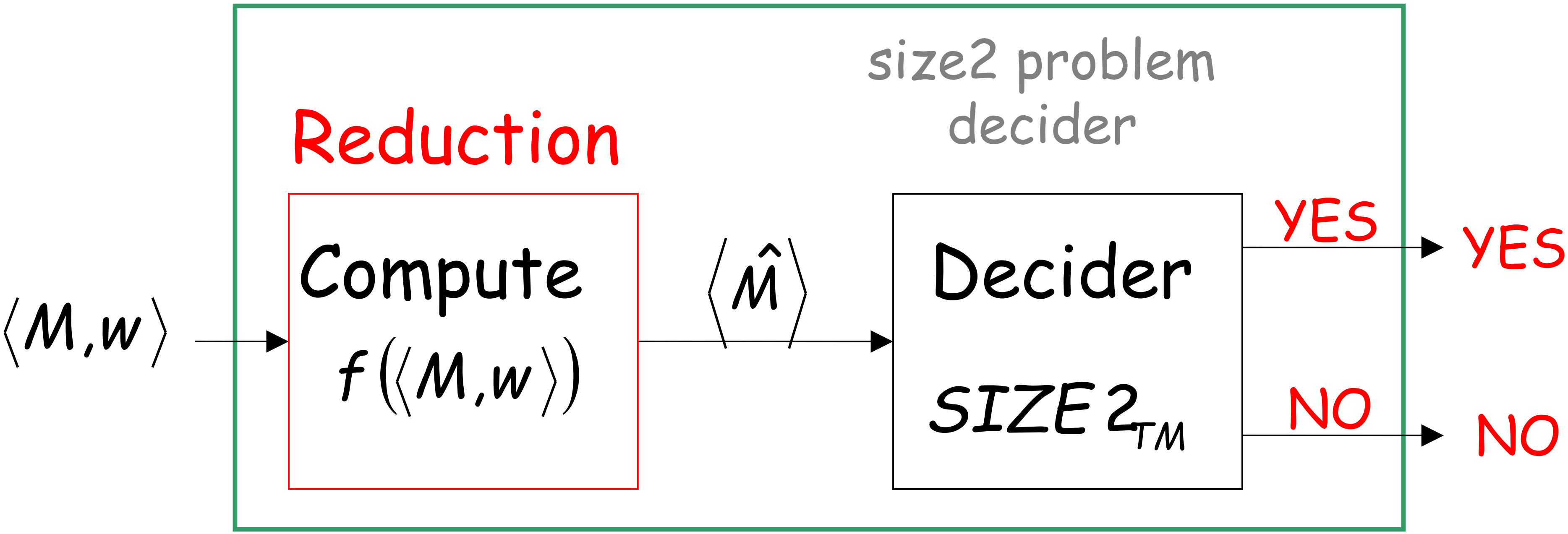
A_{TM} (membership problem)

to

$SIZE2_{TM}$ (size 2 language problem)

membership problem decider

Decider for A_{TM}



Given the reduction,
If $SIZE2_{TM}$ is decidable,
then A_{TM} is decidable

A contradiction!
since A_{TM}
is undecidable

We only need to build the reduction:

Reduction

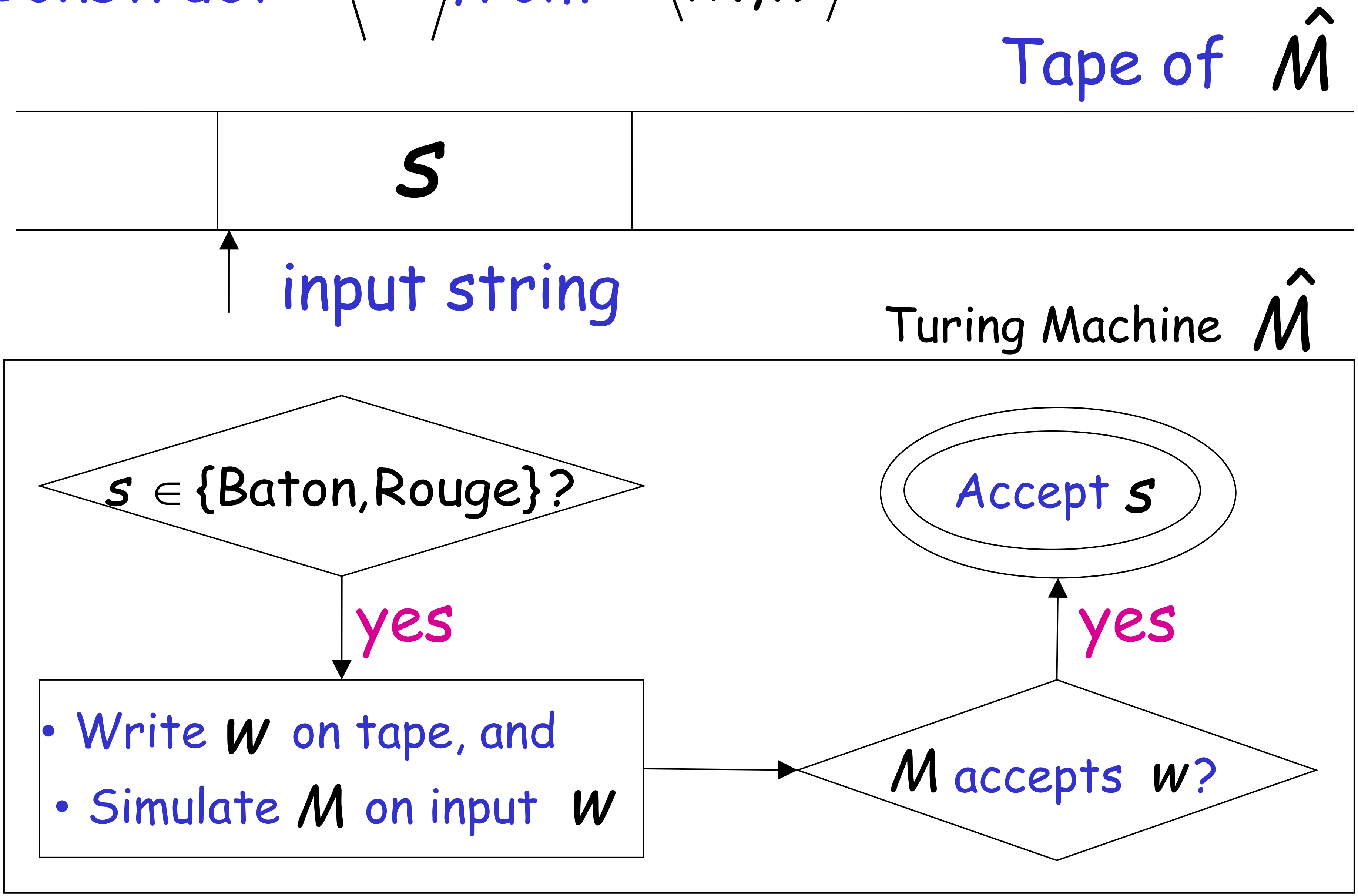
$$\langle M,w \rangle \longrightarrow \boxed{\text{Compute } f(\langle M,w \rangle)} \longrightarrow \langle \hat{M} \rangle$$

$$= f(\langle M,w \rangle)$$

So that:

$$\langle M,w \rangle \in A_{TM} \iff \langle \hat{M} \rangle \in SIZE2_{TM}$$

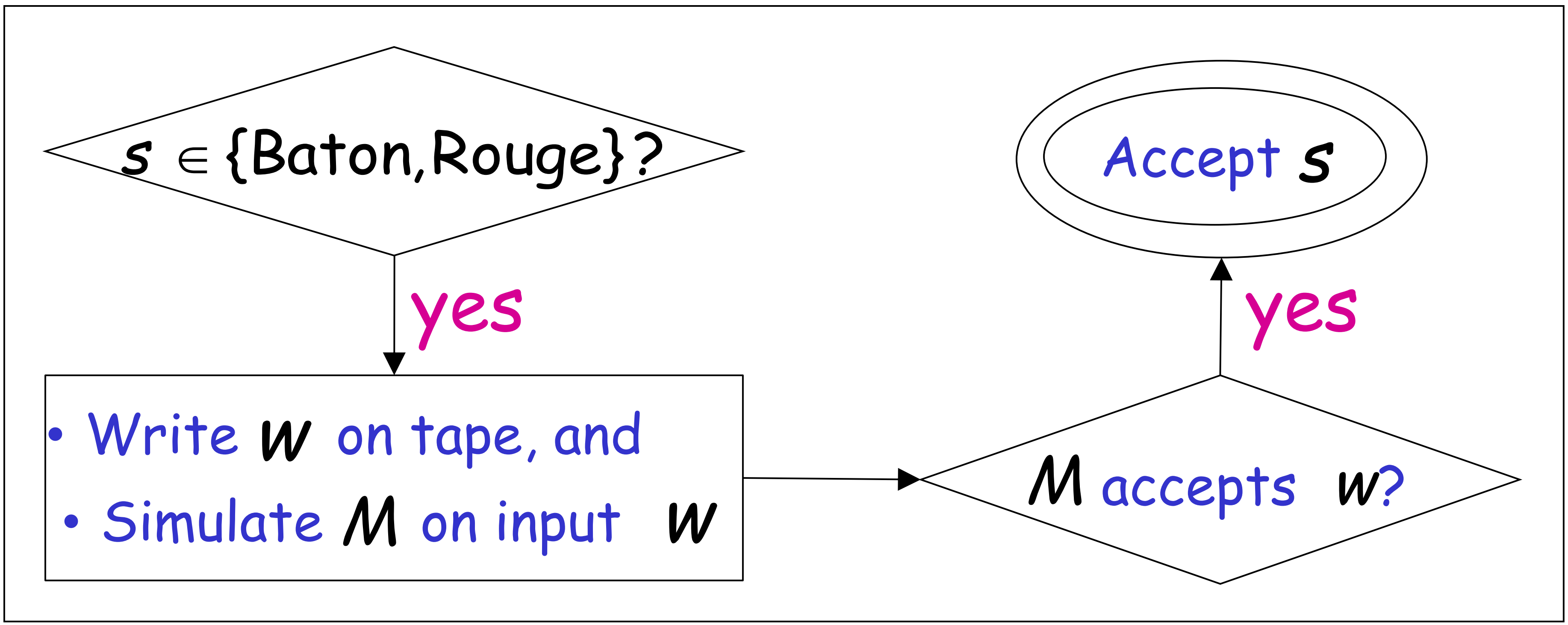
Construct $\langle \hat{M} \rangle$ from $\langle M, w \rangle$:



M accepts w $\Rightarrow$ $L(\hat{M}) = \{Baton, Rouge\}$ 2 strings

M does not accept w $\Rightarrow$ $L(\hat{M}) = \varnothing$ 0 strings

Turing Machine $\hat{M}$



Therefore:

M accepts w $\Longleftrightarrow$ $L(\hat{M})$ has size 2

Equivalently:

$$\langle M, w \rangle \in A_{TM} \Longleftrightarrow \langle \hat{M} \rangle \in SIZE2_{TM}$$

END OF PROOF

RICE's Theorem

Undecidable problems:

- L is empty?
- L is regular?
- L has size 2?

This can be generalized to all non-trivial properties of Turing-acceptable languages

Non-trivial property:

A property P possessed by some Turing-acceptable languages but not all

Example: P_1 : L is empty?

YES $L = \emptyset$

NO $L = \{\text{Louisiana}\}$

NO $L = \{\text{Baton}, \text{Rouge}\}$

More examples of non-trivial properties:

P_2 : L is regular?

YES $L = \varnothing$

YES $L = \{a^n : n \geq 0\}$

NO $L = \{a^n b^n : n \geq 0\}$

P_3 : L has size 2?

NO $L = \varnothing$

NO $L = \{\text{Louisiana}\}$

YES $L = \{\text{Baton}, \text{Rouge}\}$

Trivial property:

A property P possessed by ALL Turing-acceptable languages

Examples: P_4 : L has size at least 0?

True for all languages

P_5 : L is accepted by some Turing machine?

True for all Turing-acceptable languages