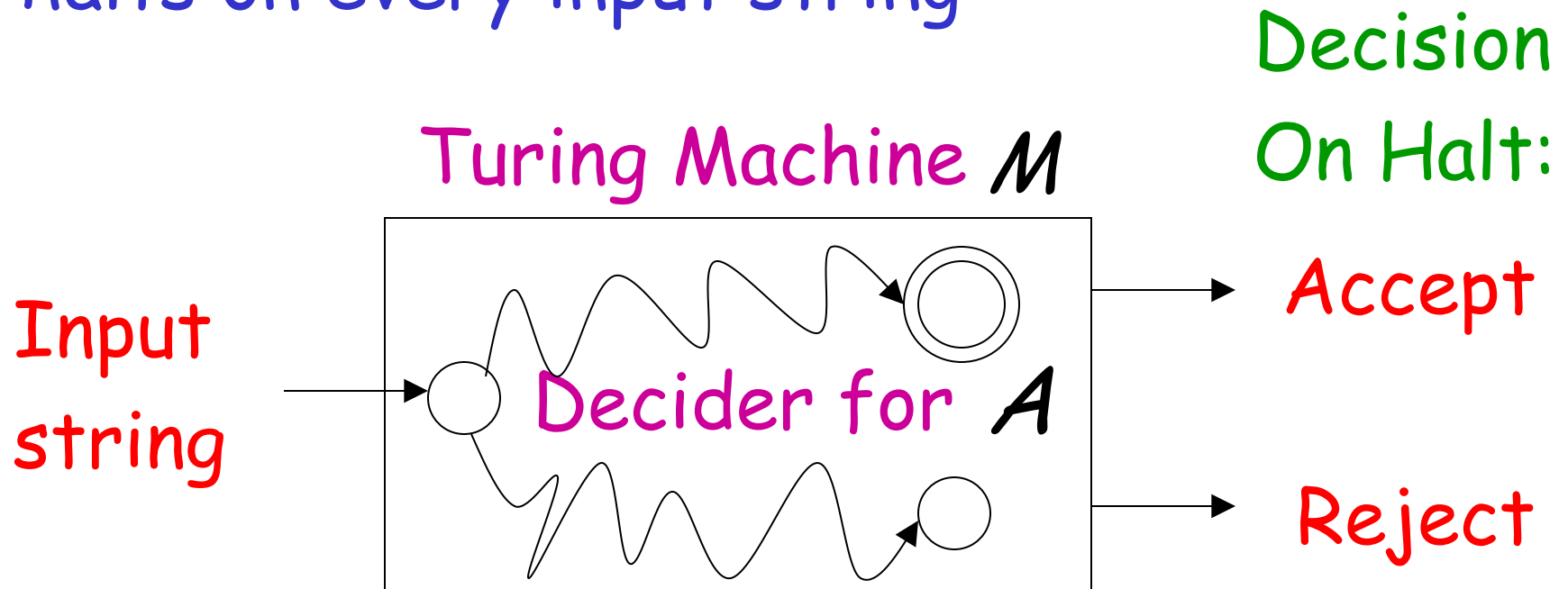


Decidability

Decidable Languages

Recall:

A language A is **decidable** (recursive), if there is a Turing machine M (decider) that accepts the language A and halts on every input string



Problem: Is number x prime?

Corresponding language:

$$PRIMES = \{1, 2, 3, 5, 7, \dots\}$$

Decider for *PRIMES* :

On input number x :

Divide x with all possible numbers
between 2 and $\sqrt{x}$

If any of them divides x

Then reject

Else accept

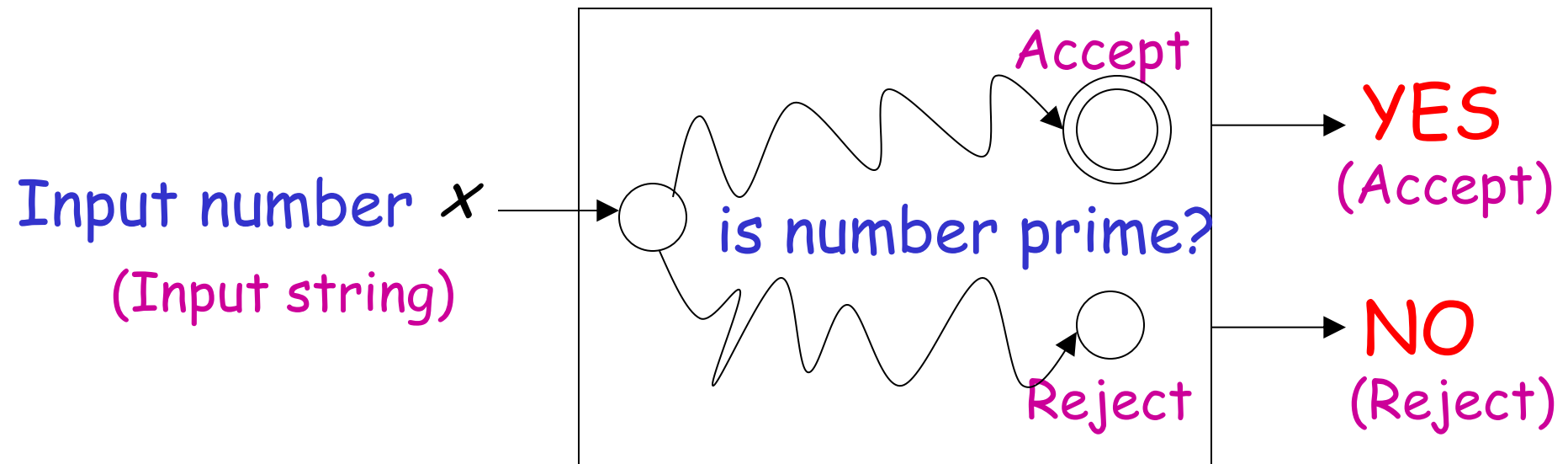
Thus, *PRIMES* is decidable

We also say that the corresponding prime number problem is solvable:

we can give an answer (positive or negative)
for every input instance of the problem

the decider for the language
solves the corresponding problem

Decider for *PRIMES*



Problem:

Does DFA M accept
the empty language $L(M) = \emptyset$?

Corresponding Language:

$EMPTY_{DFA} =$

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

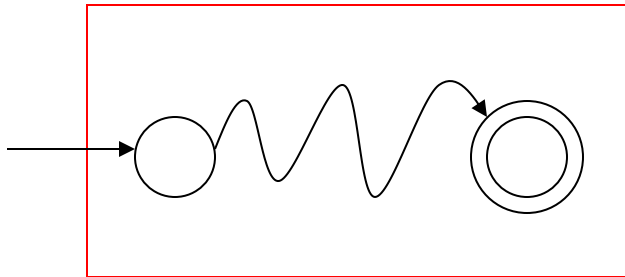
↑
Description of DFA M as a string
(For example, we can represent M as a
binary string, as we did for Turing machines)

Decider for $EMPTY_{DFA}$:

On input $\langle M \rangle$:

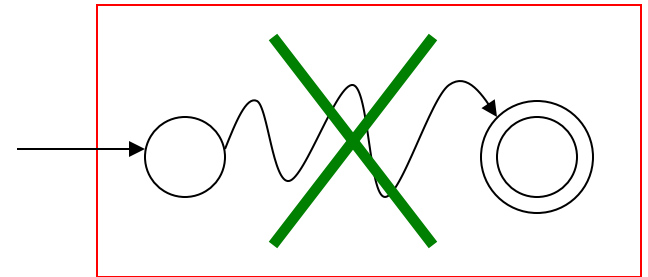
Determine whether there is a path from the initial state to any accepting state

DFA M



$$L(M) \neq \emptyset$$

DFA M



$$L(M) = \emptyset$$

Decision: Reject $\langle M \rangle$

Accept $\langle M \rangle$

Problem: Does DFA M accept a finite language?

Corresponding Language:

$FINITE_{DFA} =$

$\{\langle M \rangle : M \text{ is a DFA that accepts a finite language}\}$

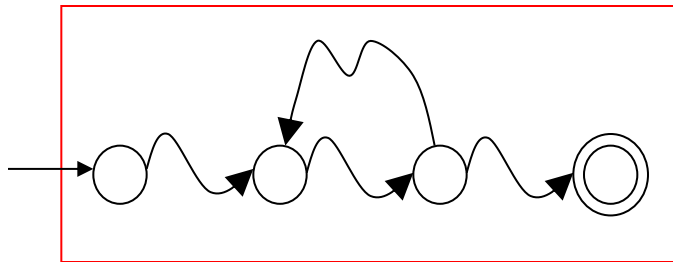
Decidable language

Decider for $FINITE_{DFA}$:

On input $\langle M \rangle$:

Check if there is a walk with cycle
from the initial state to an accepting state

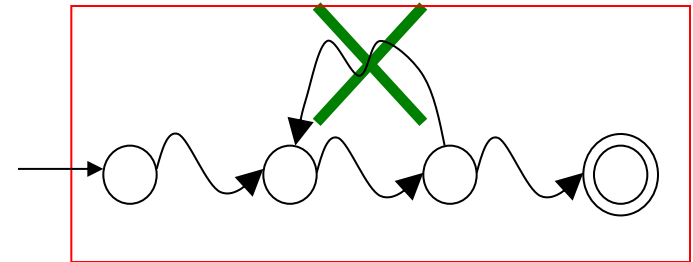
DFA M



infinite

Decision: Reject $\langle M \rangle$

DFA M



finite

Accept $\langle M \rangle$

Problem: Does DFA M accept string w ?

Corresponding Language:

$$A_{DFA} = \{ \langle M, w \rangle : M \text{ is a DFA that accepts string } w \}$$

Decidable language

Decider for A_{DFA} :

On input string $\langle M, w \rangle$:

Run DFA M on input string w

If M accepts w

Then accept $\langle M, w \rangle$ (and halt)

Else reject $\langle M, w \rangle$ (and halt)

Problem:

Do DFAs M_1 and M_2
accept the same language?

Corresponding Language:

$EQUAL_{DFA} =$

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

Decidable language

Decider for $EQUAL_{DFA}$:

On input $\langle M_1, M_2 \rangle$:

Let L_1 be the language of DFA M_1

Let L_2 be the language of DFA M_2

Construct DFA M such that:

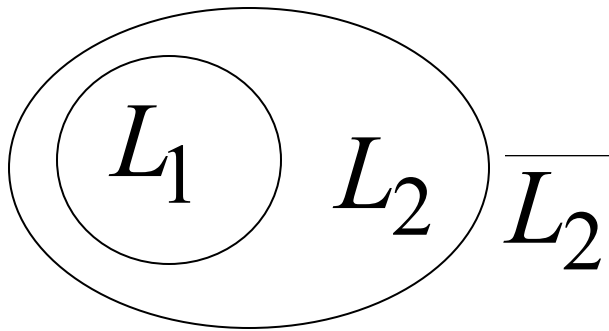
$$L(M) = (L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2)$$

(combination of DFAs)

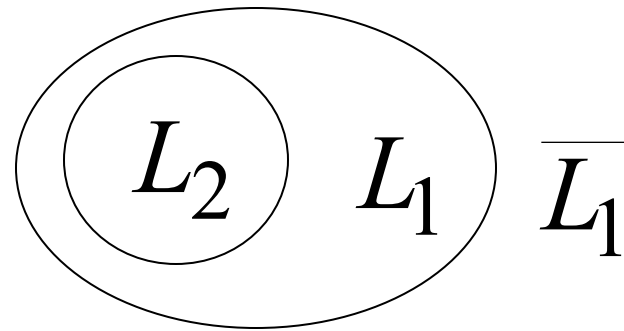
$$(L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2) = \emptyset$$



$$L_1 \cap \overline{L_2} = \emptyset \quad \text{and} \quad \overline{L_1} \cap L_2 = \emptyset$$



$$L_1 \subseteq L_2$$



$$L_2 \subseteq L_1$$



$$L_1 = L_2$$

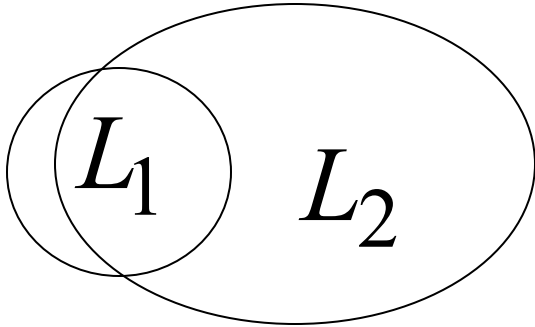
$$(L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2) \neq \emptyset$$



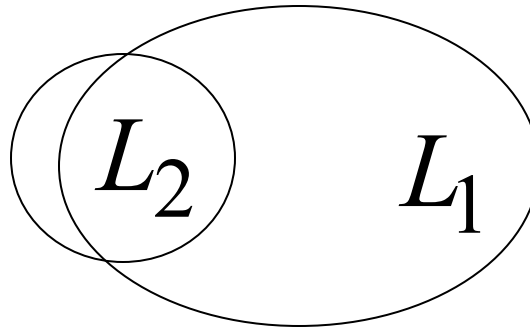
$$L_1 \cap \overline{L_2} \neq \emptyset$$

or

$$\overline{L_1} \cap L_2 \neq \emptyset$$



$$L_1 \not\subseteq L_2$$



$$L_2 \not\subseteq L_1$$



$$L_1 \neq L_2$$

Therefore, we only need
to determine whether

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

which is a solvable problem for DFAs:

$$L(M) \in \text{EMPTY}_{DFA}?$$

Undecidable Languages

undecidable language =
language is not decidable

There is no decider for the language:
there is no Turing Machine
that reaches a decision (halts)
for all input strings of the language

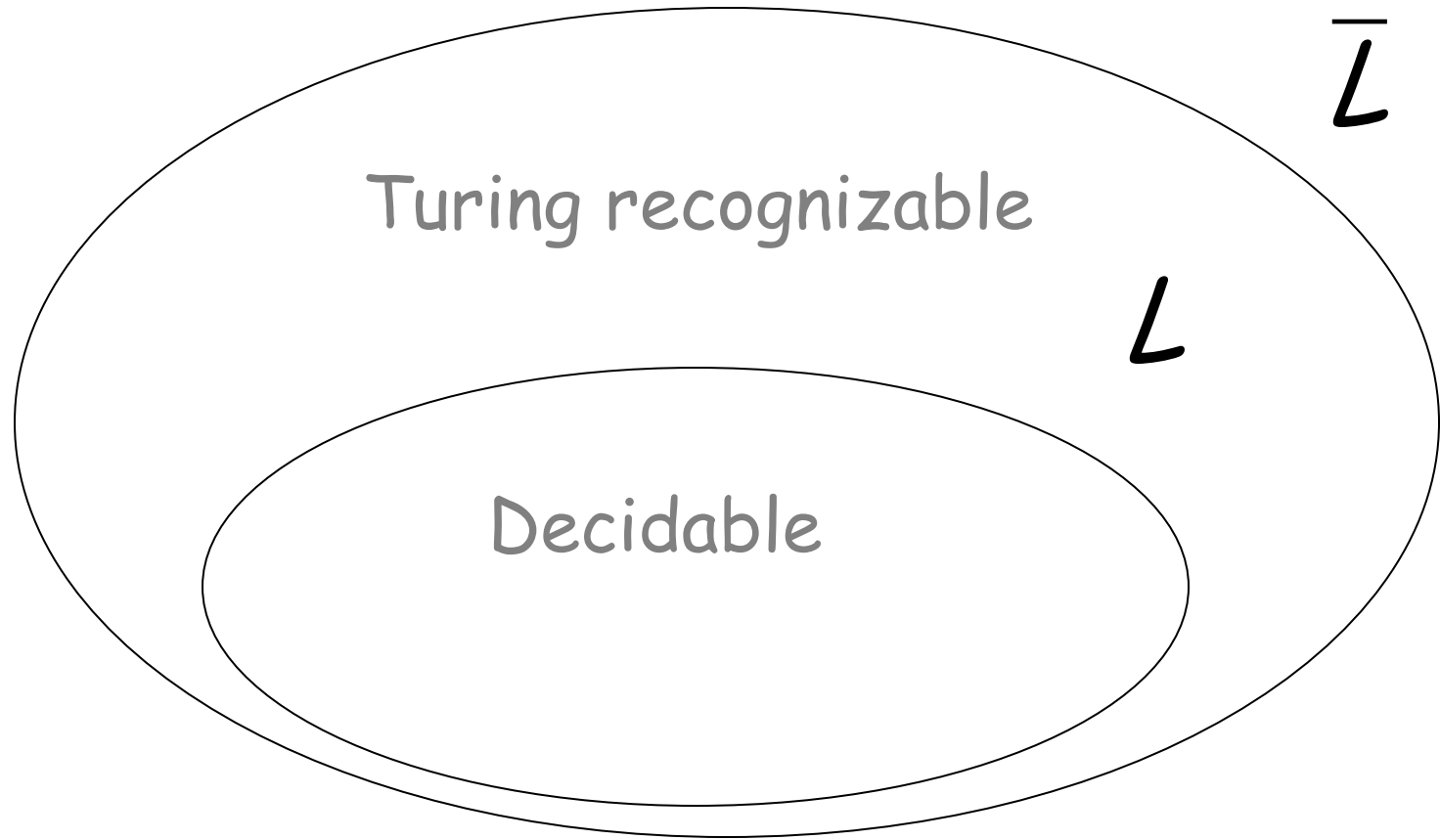
(decision may be reached for some strings)

For an **undecidable** language,
the corresponding problem is **unsolvable**:

there is no Turing Machine (Algorithm)
that gives an answer (solves the problem)
for all input instances of the problem

(answer may be given for some input instances)

We have shown before that there are undecidable languages:



L is Turing recognizable but not decidable

We will prove that two particular problems
are unsolvable:

Membership problem

Halting problem

Membership Problem

Input:

- Turing Machine M
- String w

Question: Does M accept w ?
 $w \in L(M)$?

Corresponding language:

$$A_{TM} = \{ \langle M, w \rangle : M \text{ is a Turing machine that accepts string } w \}$$

Theorem: A_{TM} is undecidable

(The membership problem is unsolvable)

Proof:

Basic idea:

We will assume that A_{TM} is decidable;

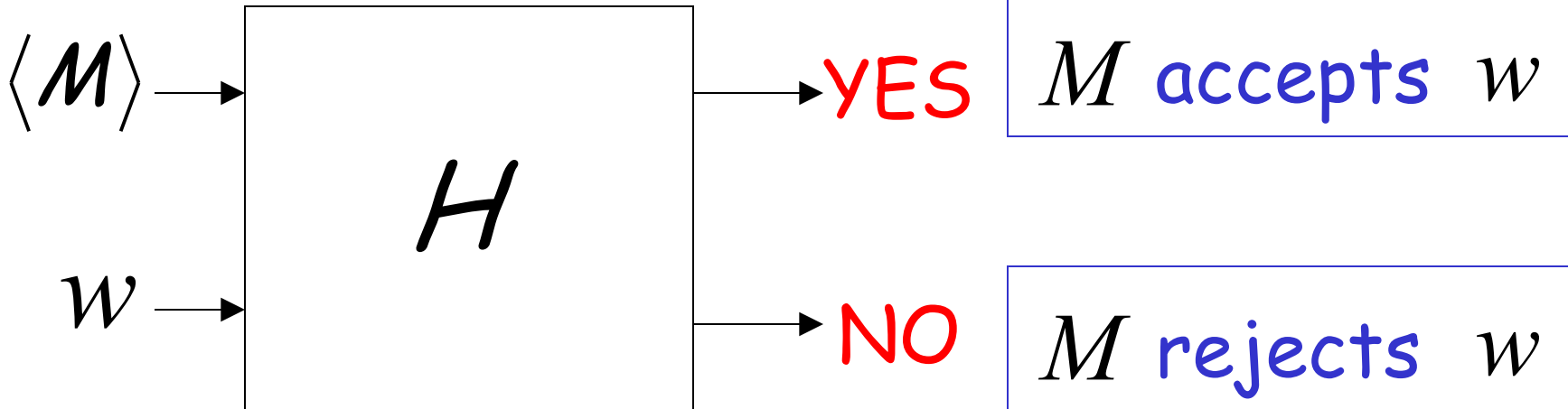
We will then prove that
every decidable language
is also Turing recognizable

A contradiction!

Suppose that A_{TM} is decidable

Input
string
 $\langle M, w \rangle$

Decider
for A_{TM}



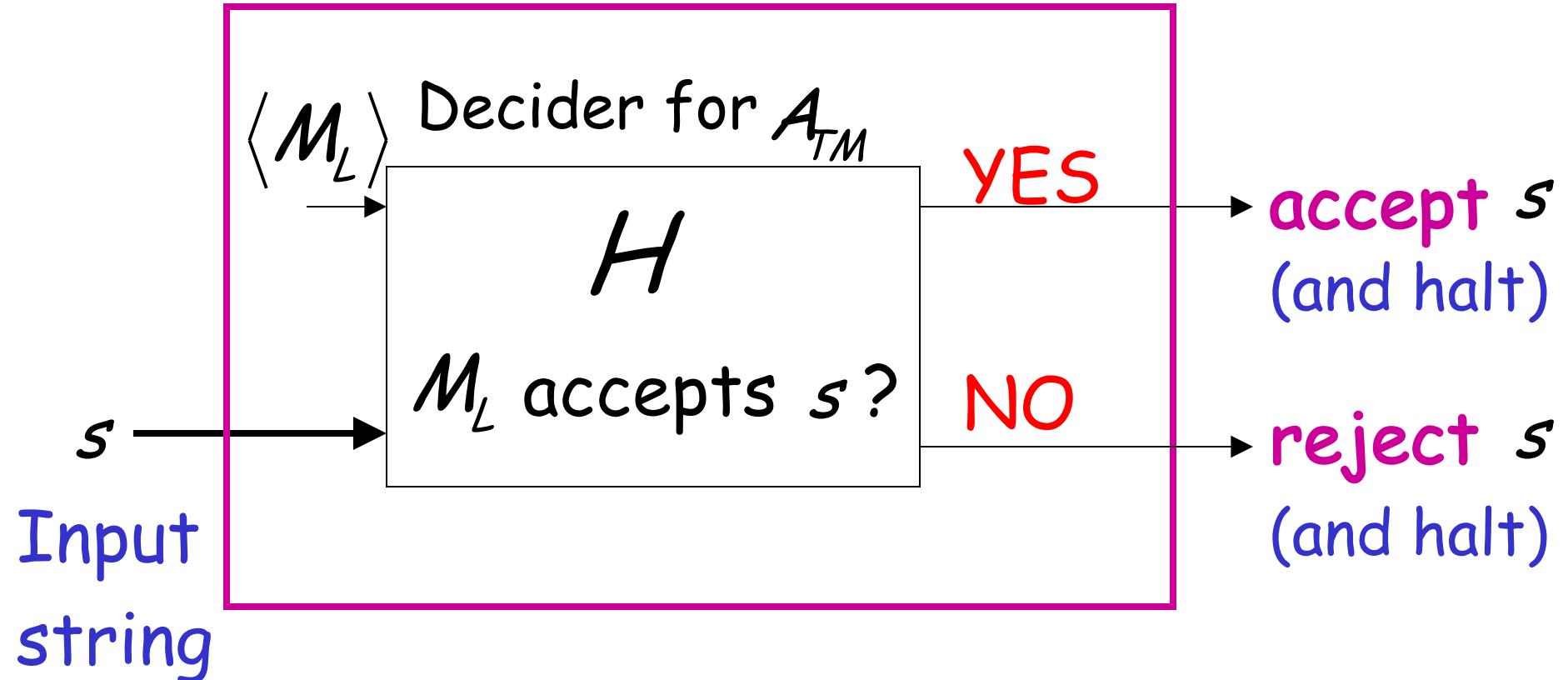
Let L be a Turing recognizable language

Let M_L be the Turing Machine that accepts L

We will prove that L is also decidable:

we will build a decider for L

Decider for L



Therefore, L is decidable

Since L is chosen arbitrarily, every
Turing recognizable language
is also decidable

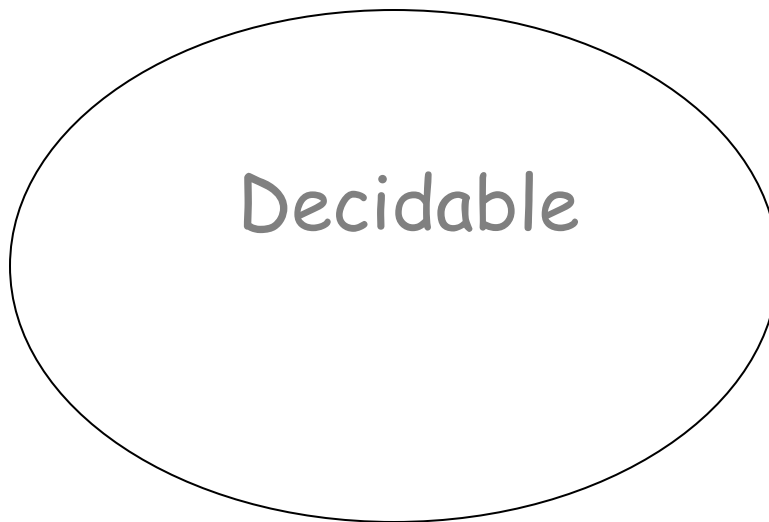
But there are Turing recognizable
languages which are not decidable

Contradiction!!!!

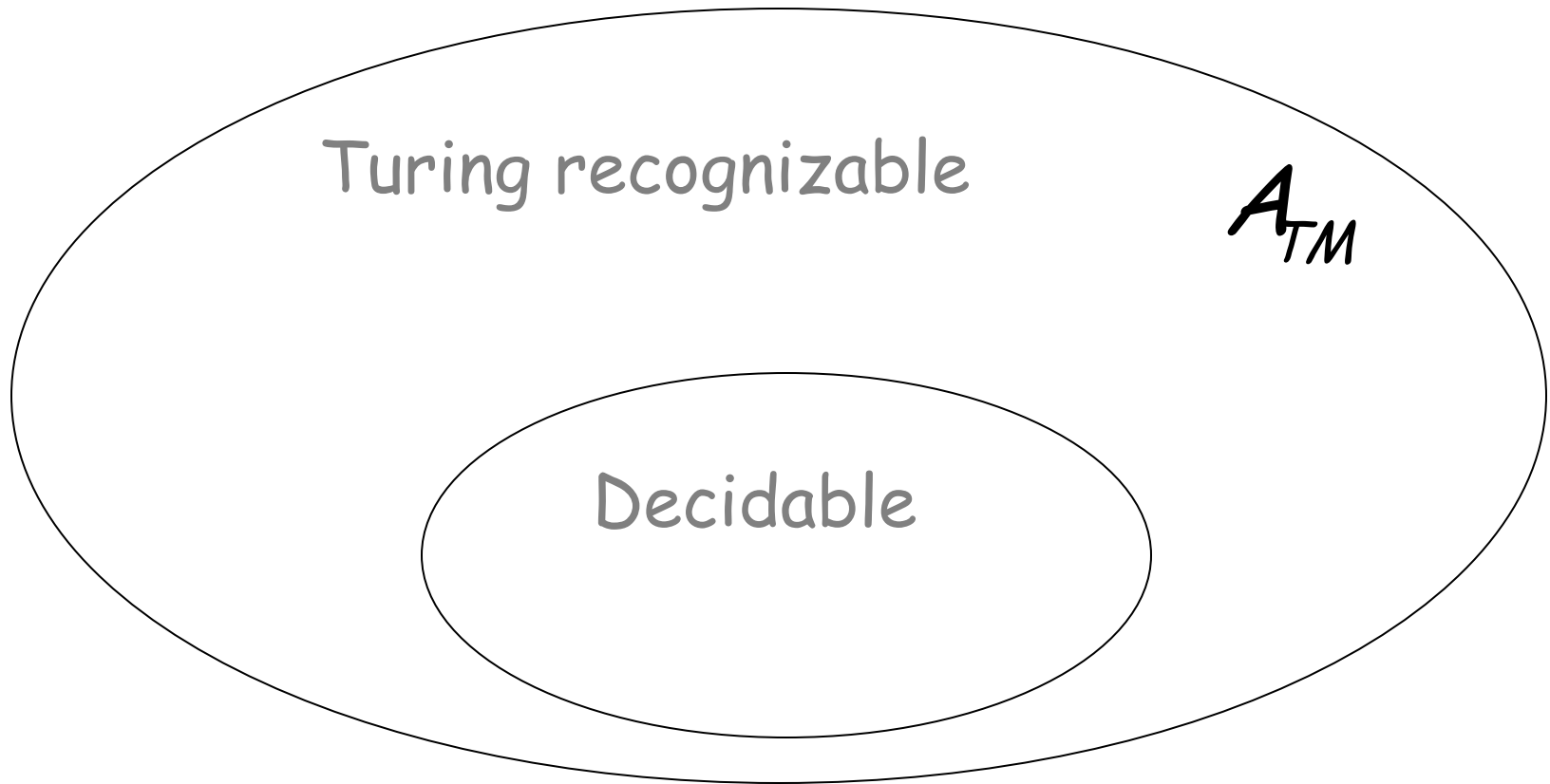
END OF PROOF

We have shown:

Undecidable A_{TM}

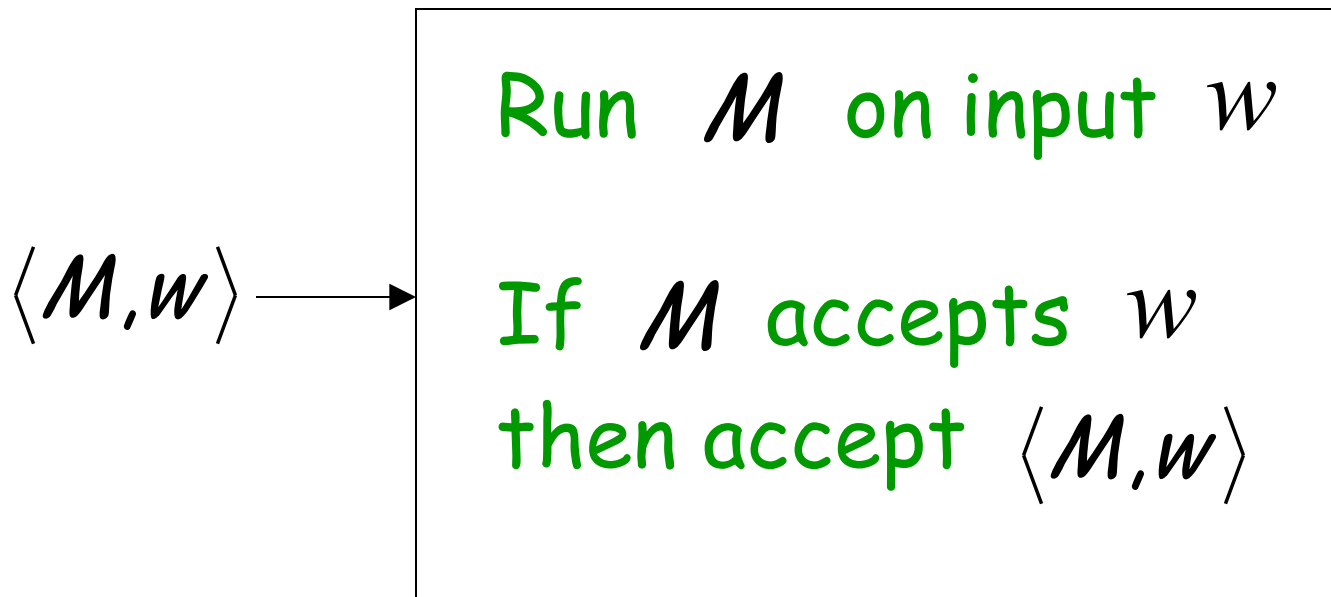


We can actually show:



A_{TM} is Turing recognizable

Turing machine that accepts A_{TM} :



Halting Problem

Input: • Turing Machine M
• String w

Question: Does M halt while
processing input string w ?

Corresponding language:

$HALT_{TM} = \{ \langle M, w \rangle : M \text{ is a Turing machine that} \\ \text{halts on input string } w \}$

Theorem: $HALT_{TM}$ is undecidable
(The halting problem is unsolvable)

Proof:

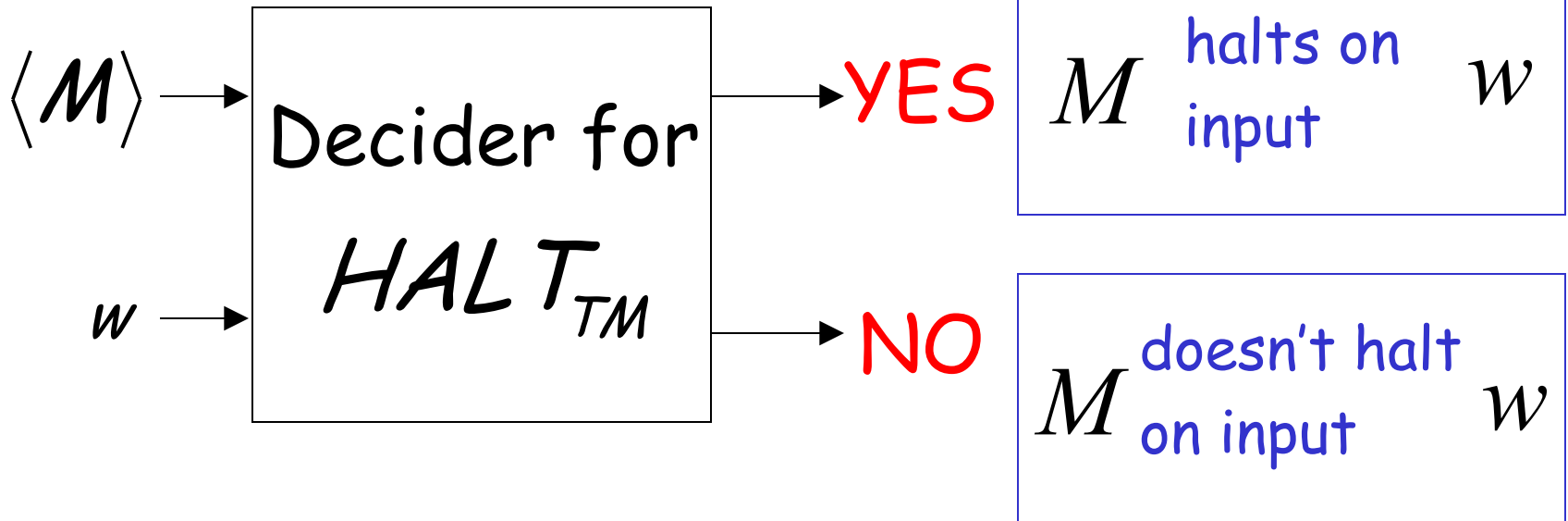
Basic idea:

Suppose that $HALT_{TM}$ is decidable;
we will prove that
every decidable language
is also Turing recognizable
A contradiction!

Suppose that $HALT_{TM}$ is decidable

Input
string

$\langle M, w \rangle$



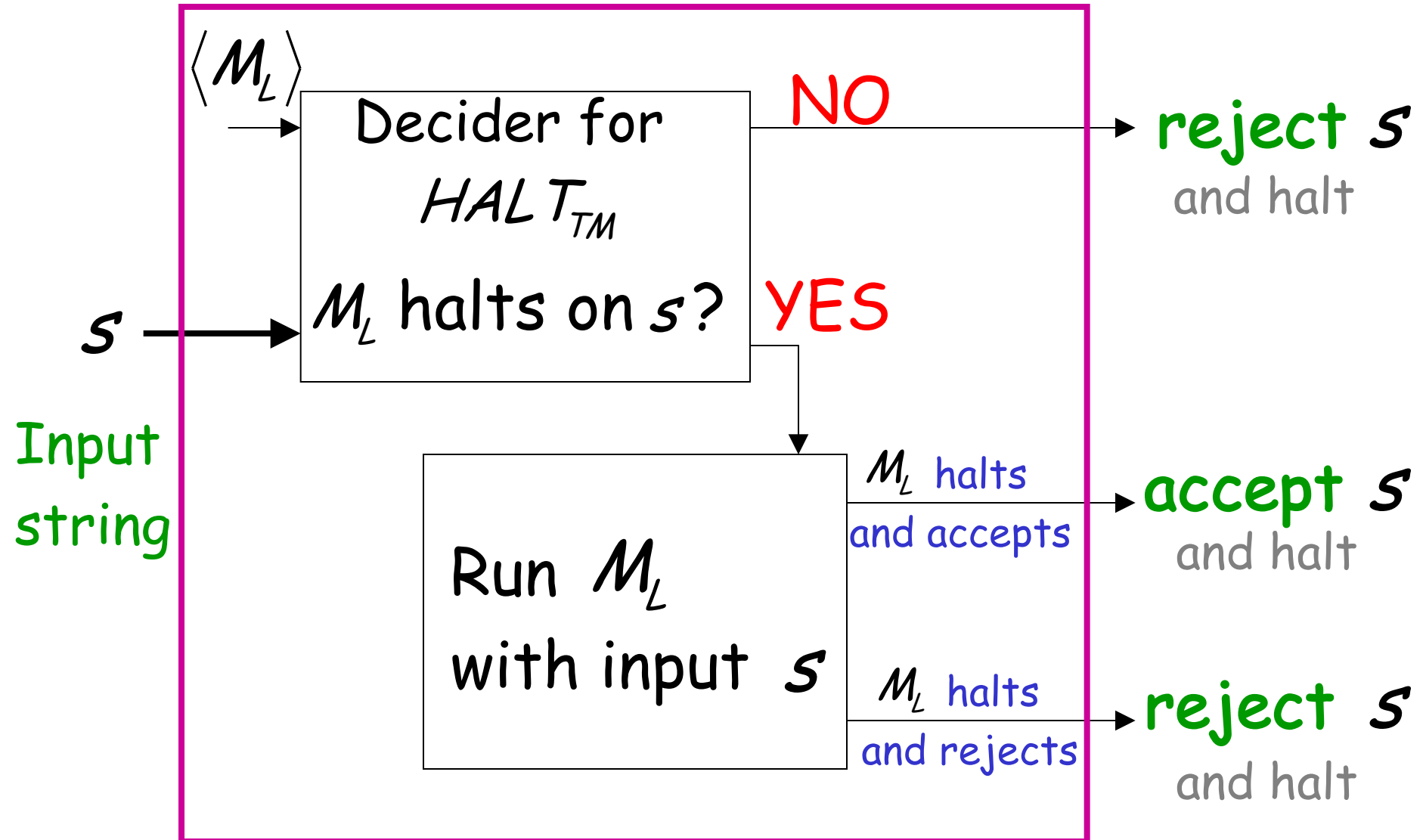
Let L be a Turing recognizable language

Let M_L be the Turing Machine that accepts L

We will prove that L is also decidable:

we will build a decider for L

Decider for L



Therefore L is decidable

Since L is chosen arbitrarily, every
Turing recognizable language
is also decidable

But there are Turing recognizable
languages which are not decidable

Contradiction!!!!

END OF PROOF

An alternative proof

Theorem: $HALT_{TM}$ is undecidable
(The halting problem is unsolvable)

Proof:

Basic idea:

Assume for contradiction that
the halting problem is decidable;

we will obtain a contradiction
using a diagonalization technique

Suppose that $HALT_{TM}$ is decidable

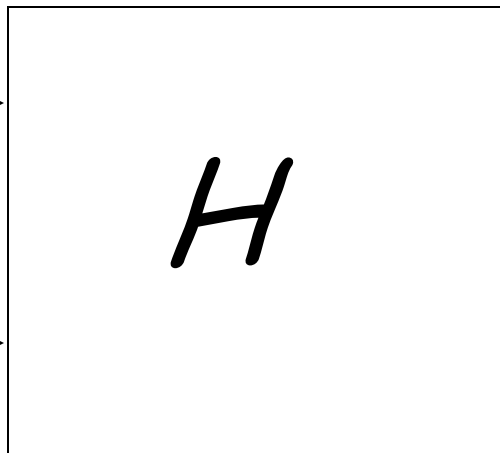
Input
string

$\langle M, w \rangle$

Decider
for $HALT_{TM}$

$\langle M \rangle$

w



YES

M halts on w

NO

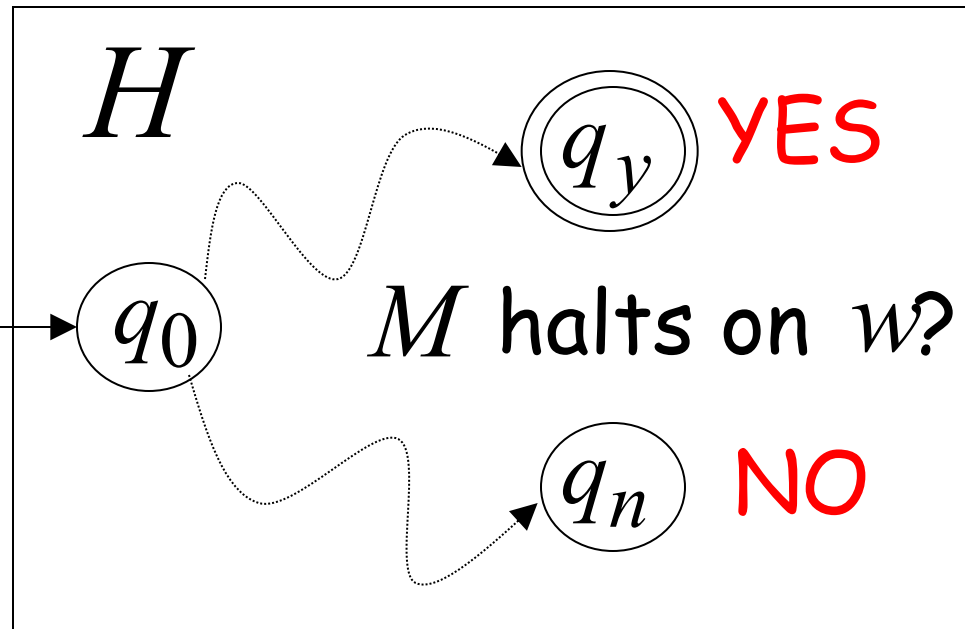
M doesn't
halt on w

Looking inside H

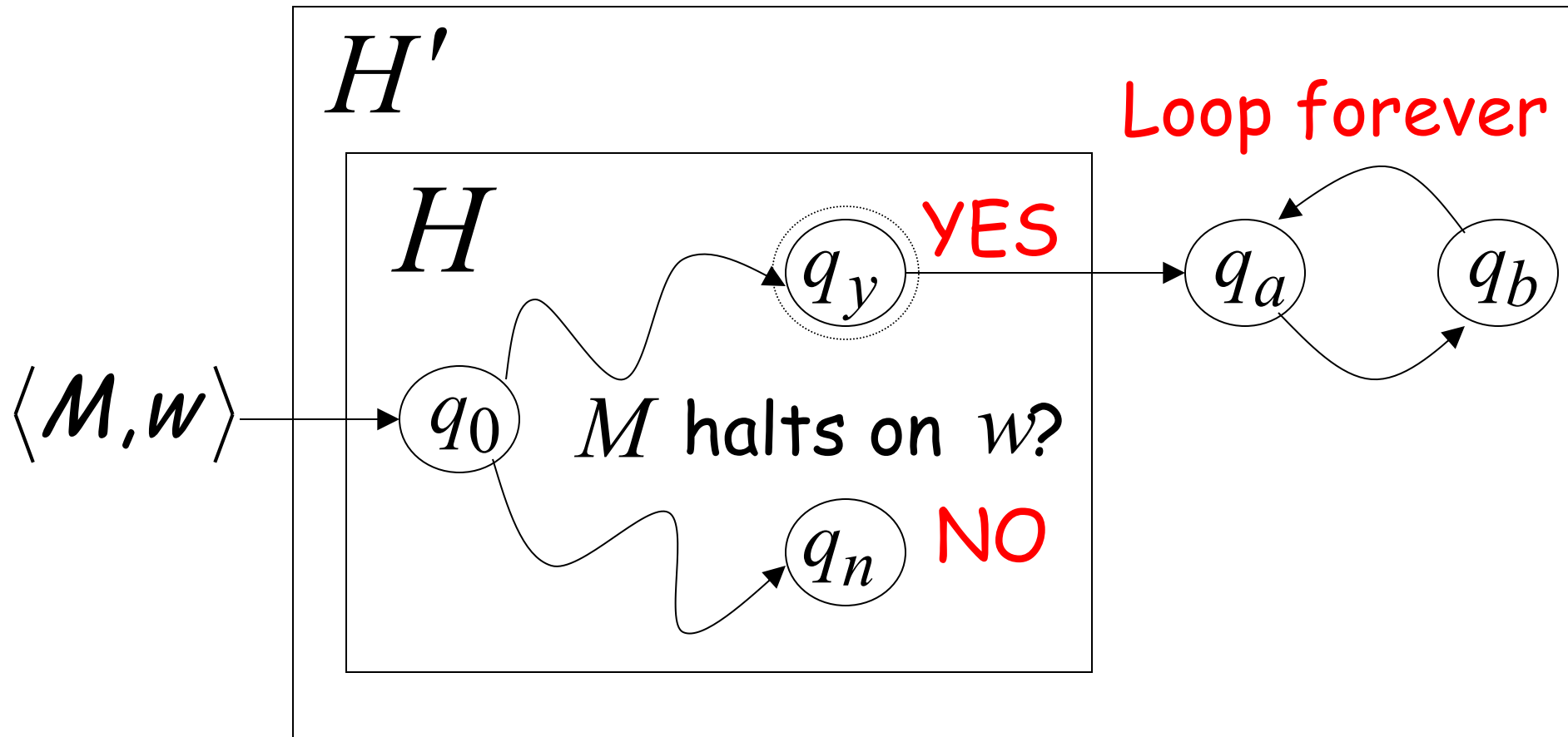
Decider for $HALT_{TM}$

Input string:

$\langle M, w \rangle$

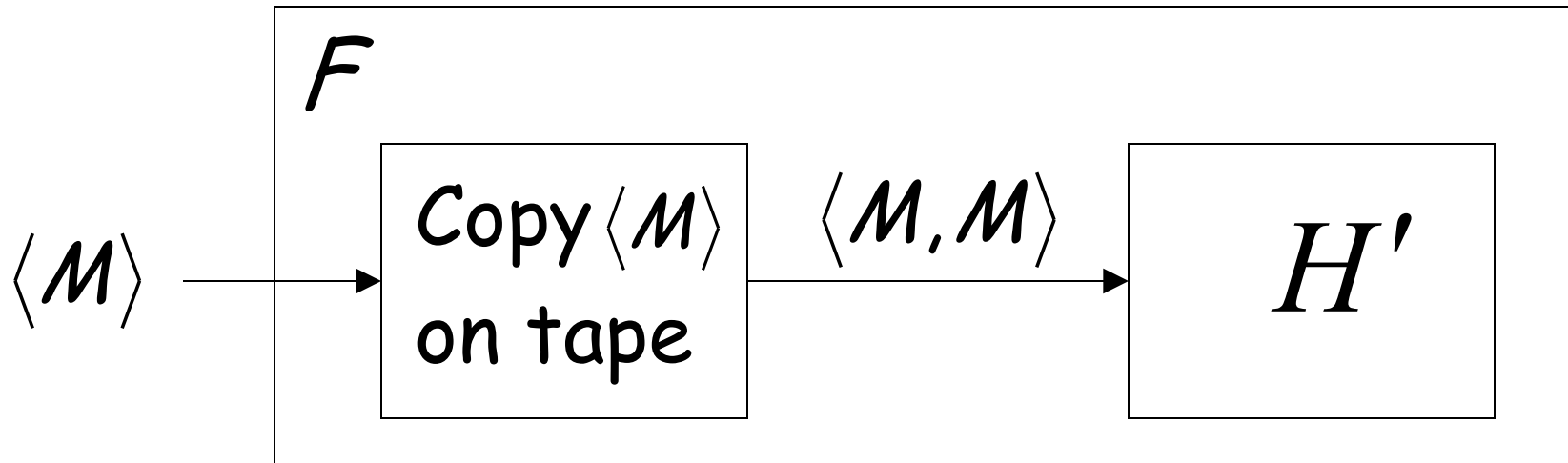


Construct machine H' :



If M halts on input w Then Loop Forever
Else Halt

Construct machine F :

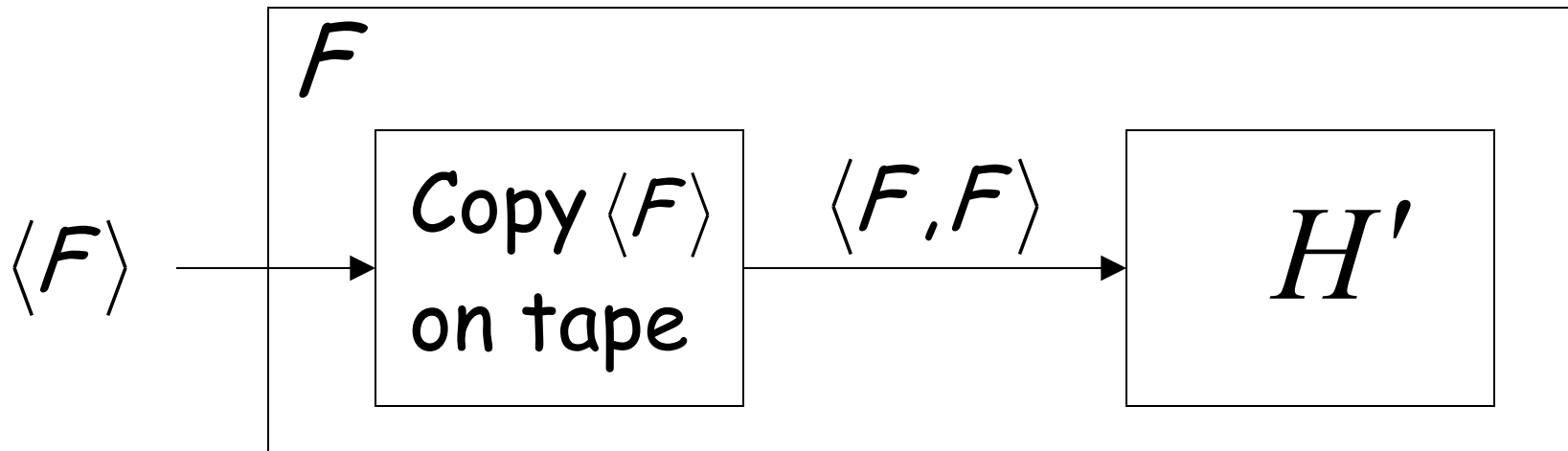


If M halts on input $\langle M \rangle$

Then loop forever

Else halt

Run F with input itself



If F halts on input $\langle F \rangle$

Then F loops forever on input $\langle F \rangle$

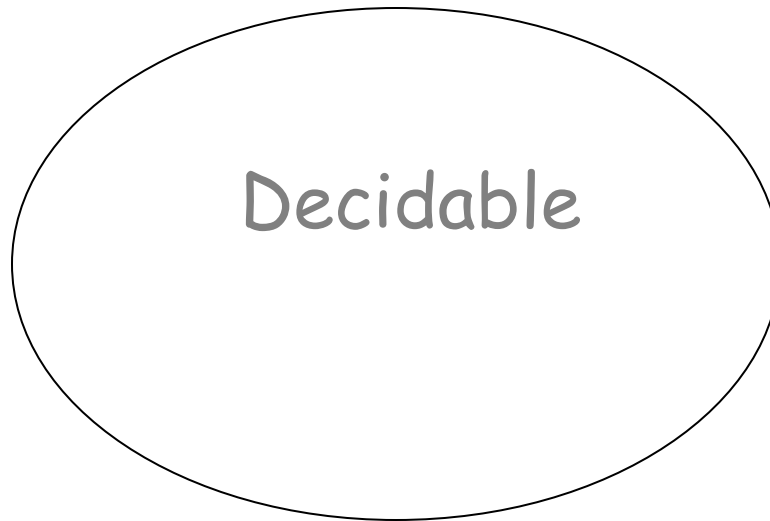
Else F halts on input $\langle F \rangle$

CONTRADICTION!!!

END OF PROOF

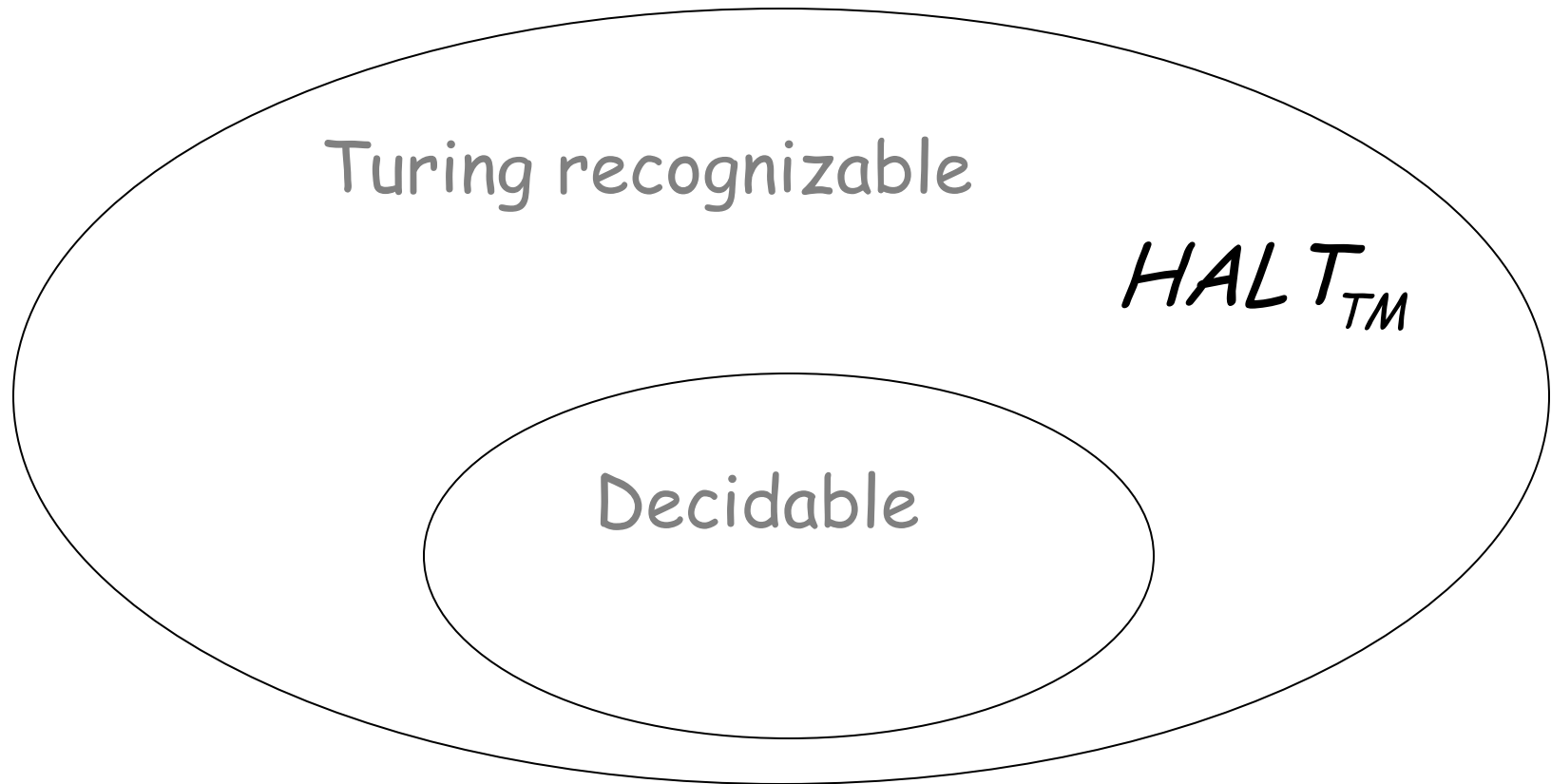
We have shown:

Undecidable $HALT_{TM}$



Decidable

We can actually show:



$HALT_{TM}$ is Turing recognizable

Turing machine that accepts $HALT_{TM}$:

