Uncountable Sets

continued...

Theorem:

Let S be an infinite countable set.

The powerset 2^S of S is uncountable

Application: Languages

Example Alphabet : $\{a,b\}$

Set of Strings:

$$S = \{a,b\}^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$$

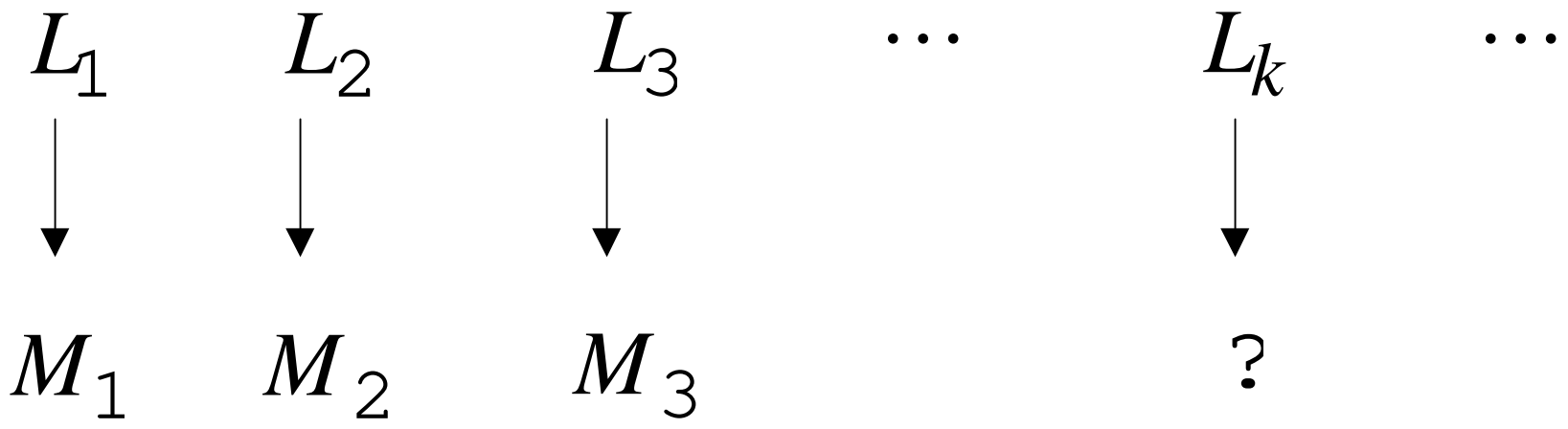
infinite and countable

Powerset: all languages

$$2^S = \{ \underbrace{\lambda}_{L_1}, \underbrace{\{\}}_{L_2}, \underbrace{\{a\}, \{a,b\}}_{L_3}, \underbrace{\{a, ab, aab\}}_{L_4}, \ldots \}$$

uncountable

Languages: Uncountable



Turing machines: Countable

There are infinitely many more languages than Turing Machines

There are some languages not accepted by Turing Machines

These languages cannot be described by algorithms

Recursively Enumerable Languages
and
Recursive Languages

Definition:

A language is **recursively enumerable** **if some Turing machine accepts it**

Let L be a recursively enumerable language

and M be the Turing Machine that accepts it

For string w :

if $w \in L$ then M halts in a final state

if $w \notin L$ then M halts in some state

or loops forever

Definition:

A language is **recursive**
if some Turing machine accepts it
and halts on any input string

In other words:

A language is recursive if there is
a membership algorithm for it

Let L be a recursive language

and M be the Turing Machine that accepts it

For string w :

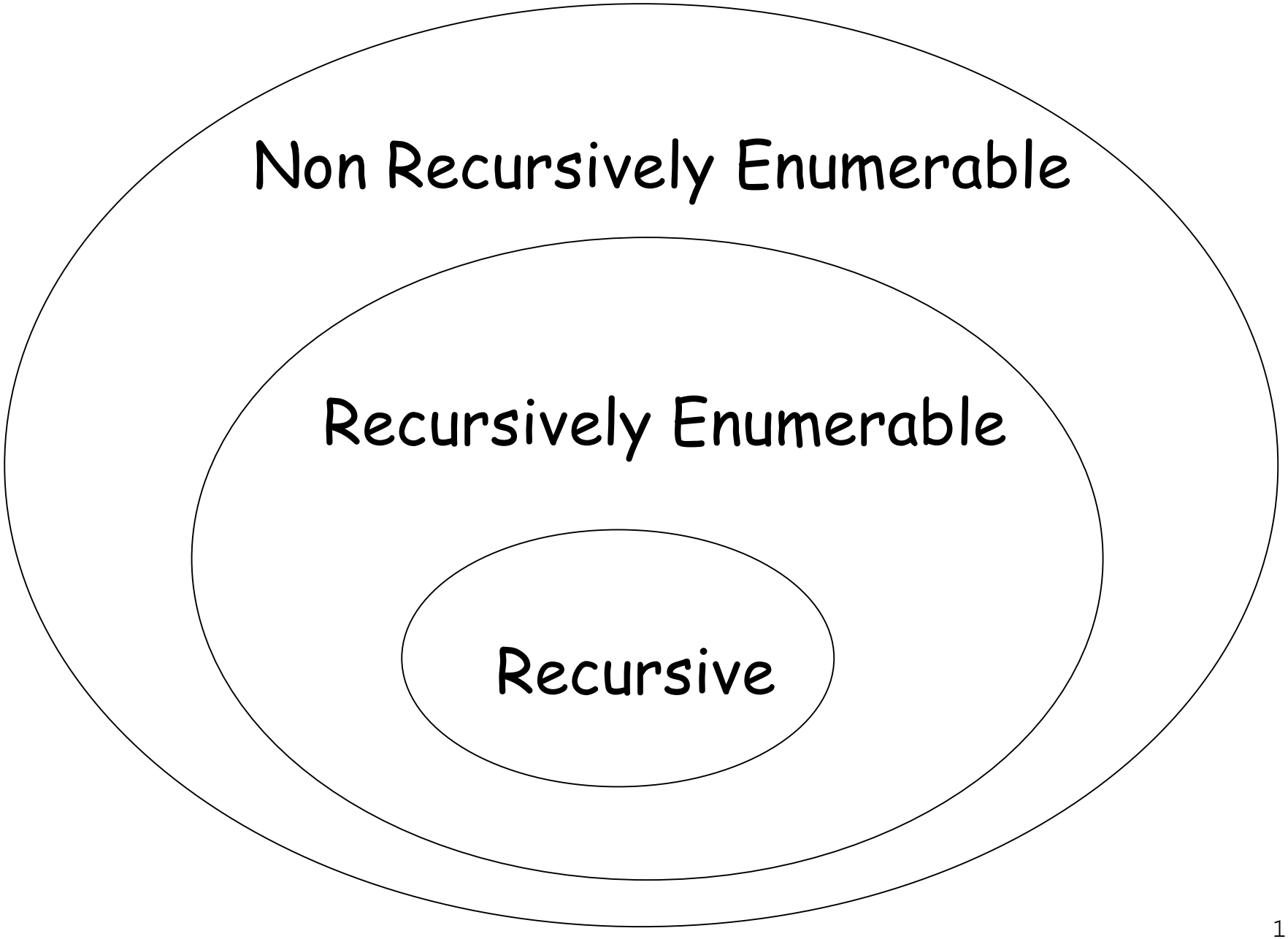
if $w \in L$ then M halts in a final state

if $w \notin L$ then M halts in a non-final state

We will prove:

1. There is a specific language
   which is not recursively enumerable

2. There is a specific language
   which is recursively enumerable
   but not recursive


Non Recursively Enumerable
Recursively Enumerable
Recursive

First we prove:

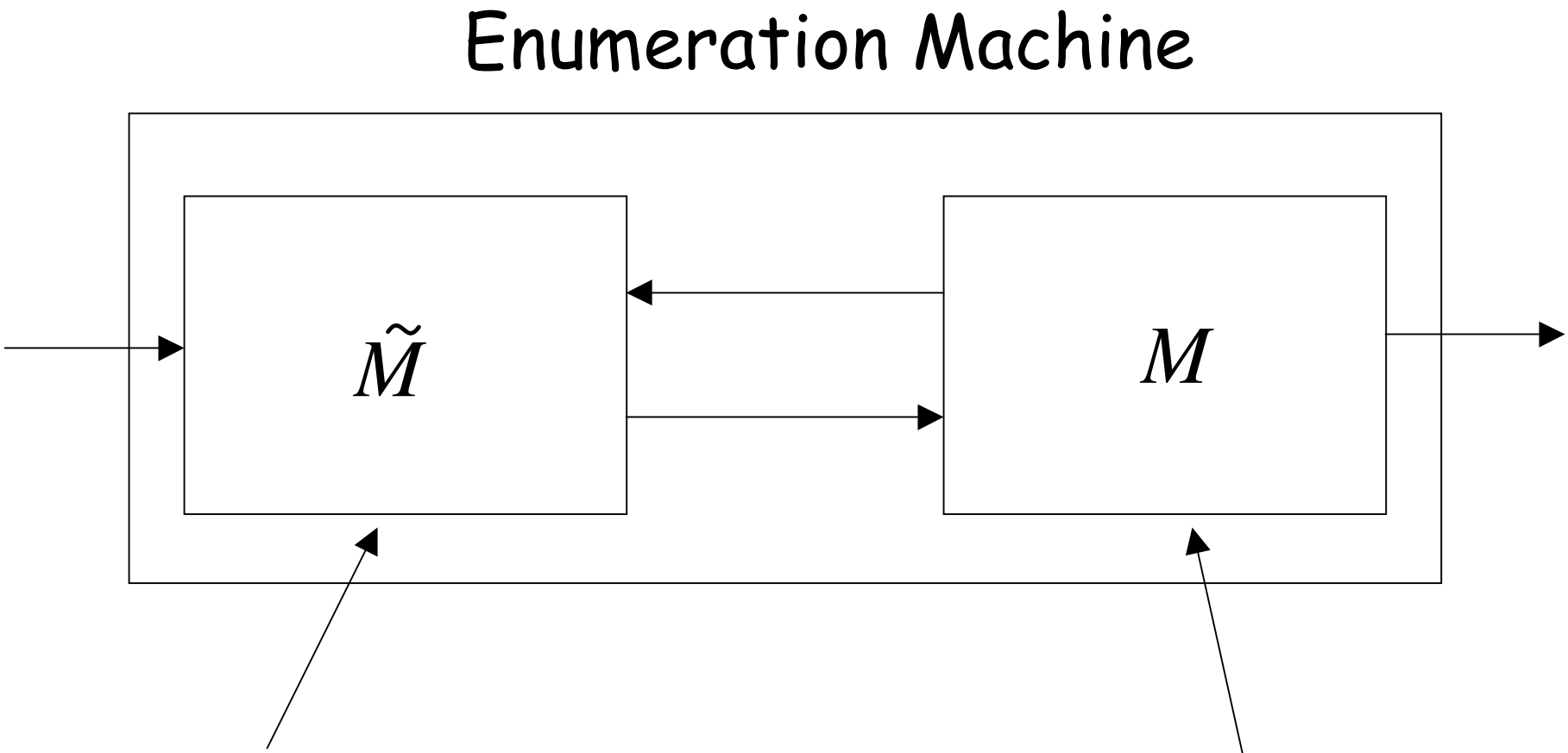
- If a language is recursive then
  there is an enumeration procedure for it

- A language is recursively enumerable
  if and only if
  there is an enumeration procedure for it

Theorem:

if a language L is recursive then
there is an enumeration procedure for it

Proof:


Enumeration Machine
$\tilde{M}$
M
Enumerates all
strings of input alphabet
Accepts L

Enumeration procedure

Repeat:

$\tilde{M}$ generates a string w

M checks if $w \in L$

YES: print w to output

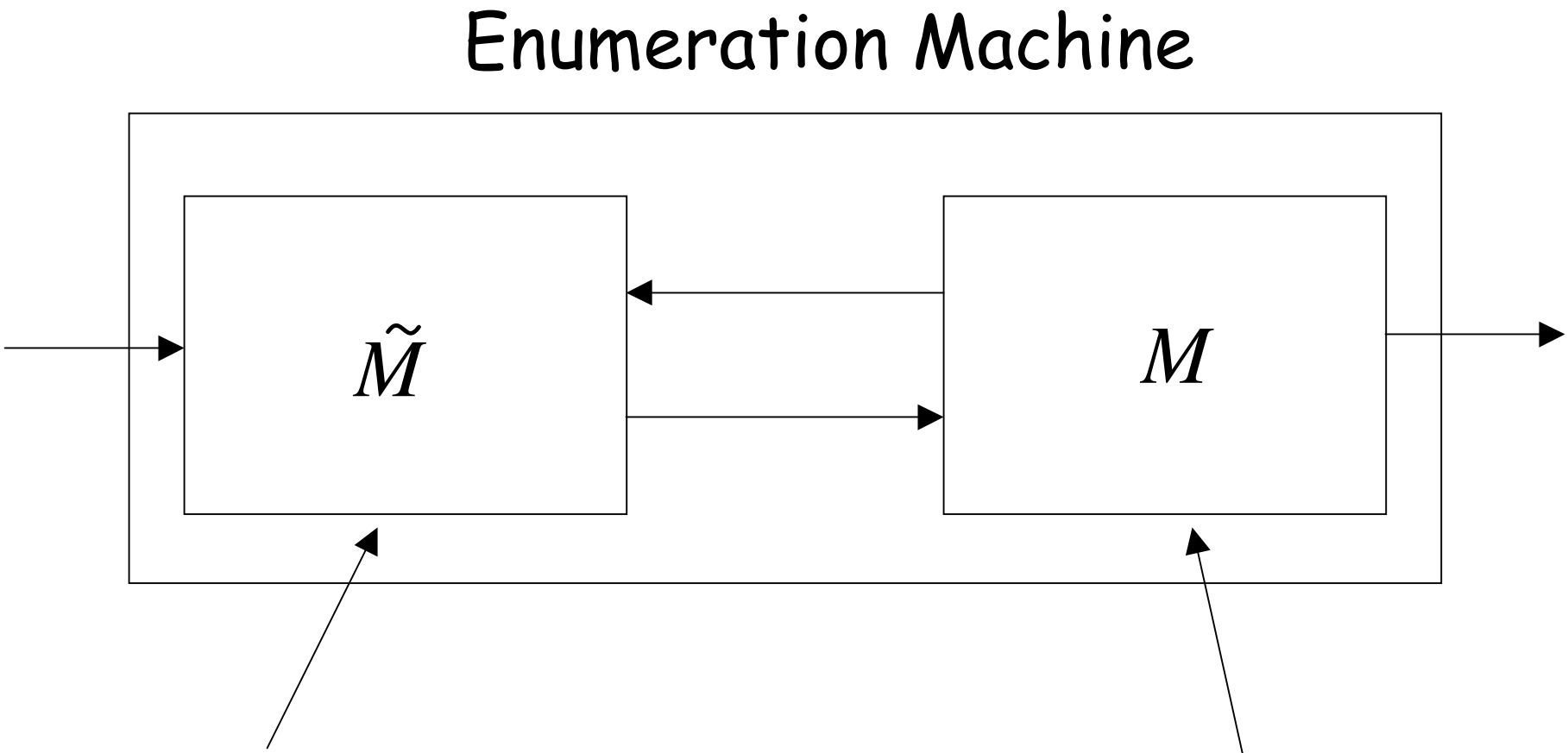
NO: ignore w

End of proof

Theorem:

if language L is recursively enumerable then there is an enumeration procedure for it

Proof:


Enumeration Machine
$\tilde{M}$
M
Enumerates all
strings of input alphabet
Accepts L

NAIVE APPROACH

Enumeration procedure

Repeat: $\tilde{M}$ generates a string w

M checks if $w \in L$

YES: print w to output

NO: ignore w

Problem: If $w \notin L$

machine M may loop forever

BETTER APPROACH

$\tilde{M}$ Generates first string w_1

M executes first step on w_1

$\tilde{M}$ Generates second string w_2

M executes first step on w_2

second step on w_1

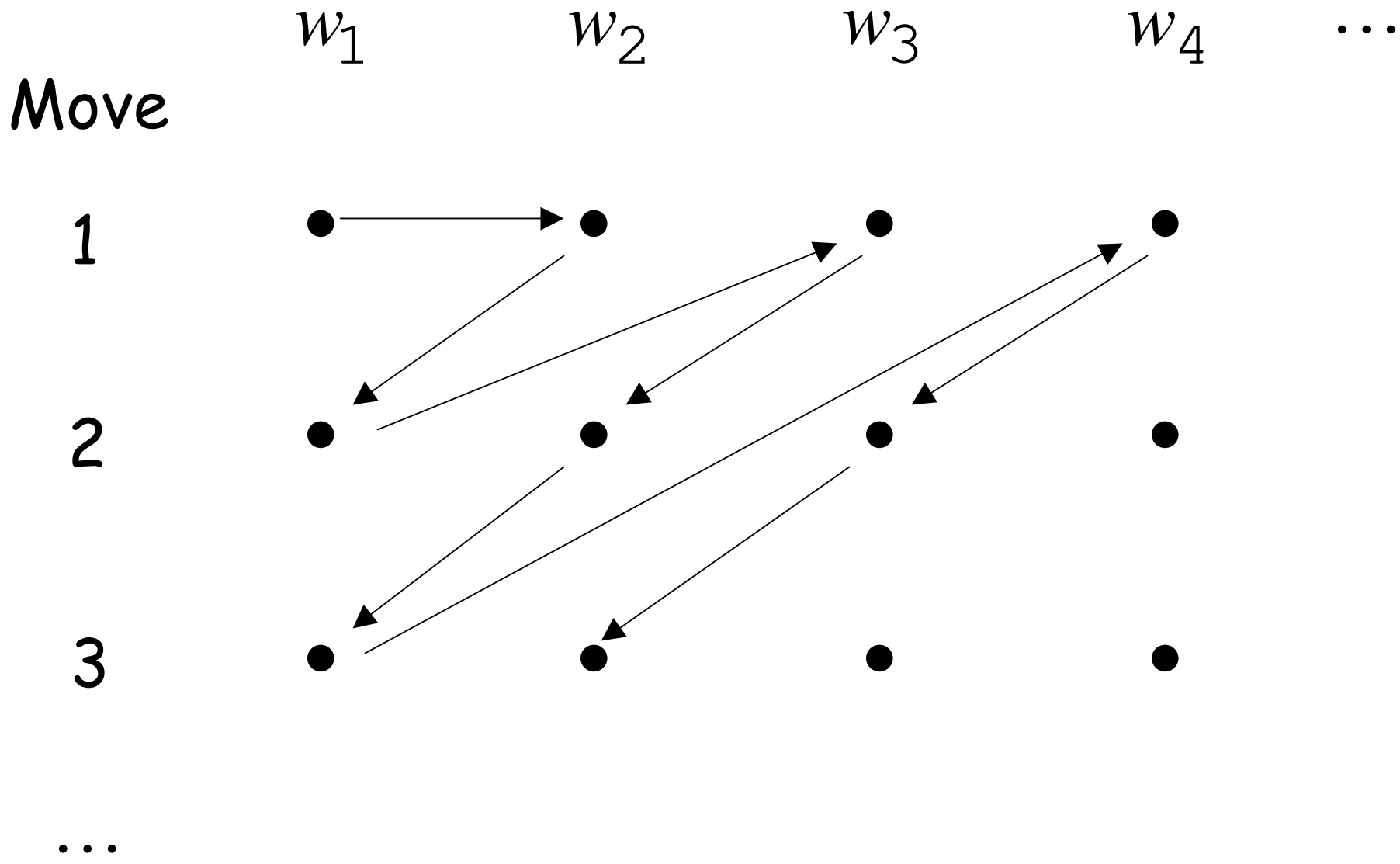
$\tilde{M}$ Generates third string w_3

M executes first step on w_3

second step on w_2

third step on w_1

And so on.....


w_1
w_2
w_3
w_4
...
Move
1
2
3
...

If for string w

machine M halts in a final state

then it prints w on the output

End of proof

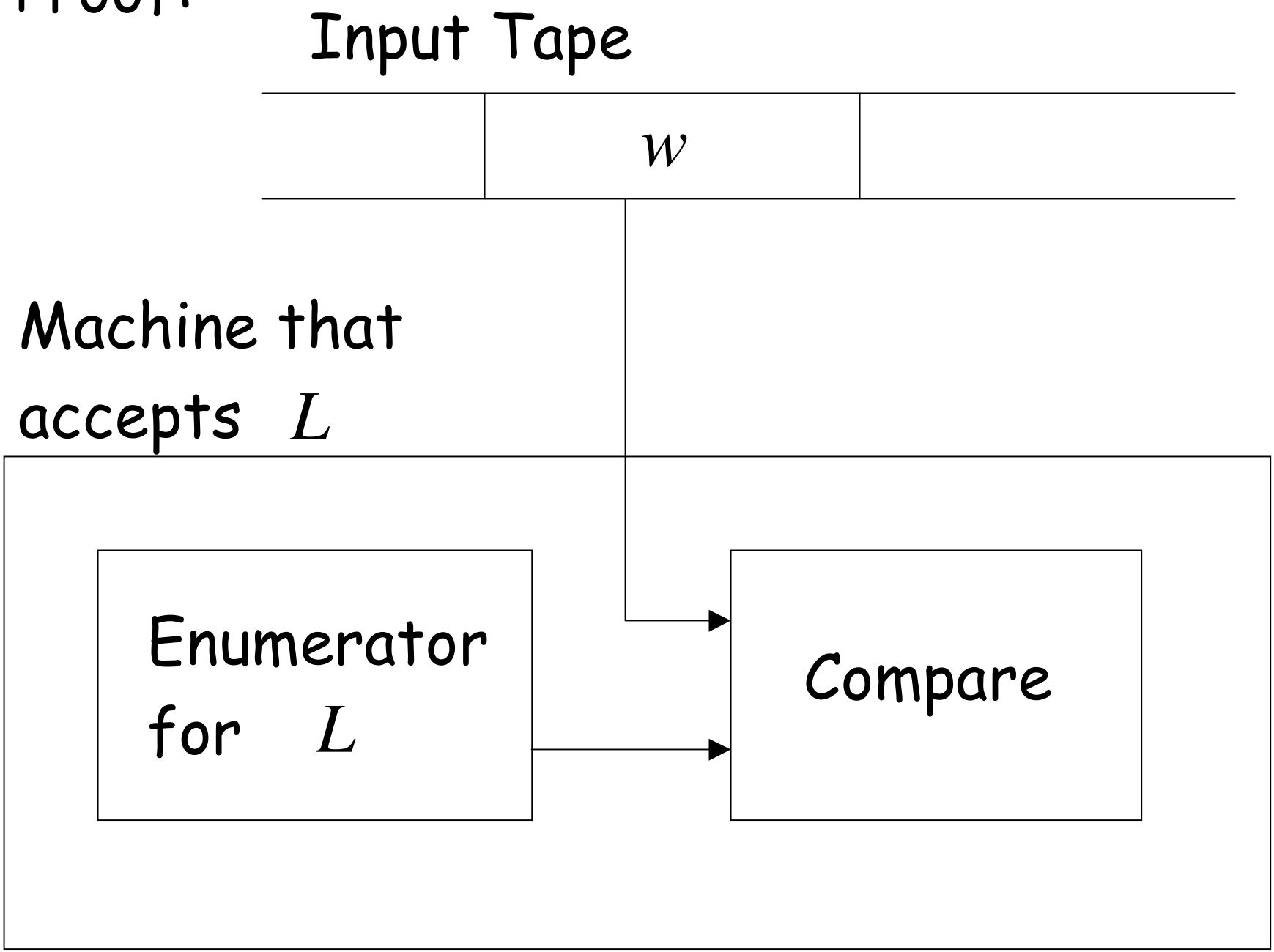
Theorem:

If for language L

there is an enumeration procedure

then L is recursively enumerable

Proof:


Input Tape
w
Machine that accepts L
Enumerator for L
Compare

Turing machine that accepts L

For input string w

Repeat:

- Using the enumerator,
  generate the next string of L

- Compare generated string with w

  If same, accept and exit loop

End of proof

Question:

This is not a membership algorithm.

Why?

Answer:

The enumeration procedure

may not produce strings in proper order

We have proven:

A language is recursively enumerable
if and only if
there is an enumeration procedure for it

A Language which is not Recursively Enumerable

We want to find a language that is not Recursively Enumerable

This language is not accepted by any Turing Machine

Consider alphabet $\{a\}$

Strings: $a, aa, aaa, aaaa, \ldots$

$$a^1 \quad a^2 \quad a^3 \quad a^4 \quad \ldots$$

Consider Turing Machines
that accept languages over alphabet $\{a\}$

They are countable:

$$M_1, M_2, M_3, M_4, \ldots$$

Example language accepted by M_i

$$L(M_i) = \{aa, aaaa, aaaaaa\}$$

$$L(M_i) = \{a^2, a^4, a^6\}$$

Alternative representation

	a^1	a^2	a^3	a^4	a^5	a^6	a^7	$\cdots$
$L(M_i)$	0	1	0	1	0	1	0	$\cdots$

	a^1	a^2	a^3	a^4	$\cdots$
$L(M_1)$	0	1	0	1	$\cdots$
$L(M_2)$	1	0	0	1	$\cdots$
$L(M_3)$	0	1	1	1	$\cdots$
$L(M_4)$	0	0	0	1	$\cdots$

Consider the language

$$L = \{a^i : a^i \in L(M_i)$$

L consists from the 1's in the diagonal

	a^1	a^2	a^3	a^4	$\cdots$
$L(M_1)$	0	1	0	1	$\cdots$
$L(M_2)$	1	0	0	1	$\cdots$
$L(M_3)$	0	1	1	1	$\cdots$
$L(M_4)$	0	0	0	1	$\cdots$

$$L = \{a^3, a^4, \ldots\}$$

Consider the language $\overline{L}$

$$L = \{a^i : a^i \in L(M_i)\}$$

$$\overline{L} = \{a^i : a^i \notin L(M_i)\}$$

$\overline{L}$ consists from the 0's in the diagonal

	a^1	a^2	a^3	a^4	$\cdots$
$L(M_1)$	0	1	0	1	$\cdots$
$L(M_2)$	1	0	0	1	$\cdots$
$L(M_3)$	0	1	1	1	$\cdots$
$L(M_4)$	0	0	0	1	$\cdots$

$$\overline{L} = \{a^1, a^2, \ldots\}$$

Theorem:

Language $\overline{L}$ is not recursively enumerable

Proof:

Assume for contradiction that

$\overline{L}$ is recursively enumerable

There must exist some machine M_k that accepts $\overline{L}$

$$L(M_k) = \overline{L}$$

	a^1	a^2	a^3	a^4	$\cdots$
$L(M_1)$	0	1	0	1	$\cdots$
$L(M_2)$	1	0	0	1	$\cdots$
$L(M_3)$	0	1	1	1	$\cdots$
$L(M_4)$	0	0	0	1	$\cdots$

Question: $M_k = M_1$?

	a^1	a^2	a^3	a^4	$\cdots$
$L(M_1)$	0	1	0	1	$\cdots$
$L(M_2)$	1	0	0	1	$\cdots$
$L(M_3)$	0	1	1	1	$\cdots$
$L(M_4)$	0	0	0	1	$\cdots$

Answer: $M_k \neq M_1$

$a^1 \in L(M_k)$

$a^1 \notin L(M_1)$

	a^1	a^2	a^3	a^4	$\cdots$
$L(M_1)$	0	1	0	1	$\cdots$
$L(M_2)$	1	0	0	1	$\cdots$
$L(M_3)$	0	1	1	1	$\cdots$
$L(M_4)$	0	0	0	1	$\cdots$

Question: $M_k = M_2$?

	a^1	a^2	a^3	a^4	$\cdots$
$L(M_1)$	0	1	0	1	$\cdots$
$L(M_2)$	1	0	0	1	$\cdots$
$L(M_3)$	0	1	1	1	$\cdots$
$L(M_4)$	0	0	0	1	$\cdots$

Answer: $M_k \neq M_2$

$a^2 \in L(M_k)$

$a^2 \notin L(M_2)$

	a^1	a^2	a^3	a^4	$\cdots$
$L(M_1)$	0	1	0	1	$\cdots$
$L(M_2)$	1	0	0	1	$\cdots$
$L(M_3)$	0	1	1	1	$\cdots$
$L(M_4)$	0	0	0	1	$\cdots$

Question: $M_k = M_3$?

	a^1	a^2	a^3	a^4	$\cdots$
$L(M_1)$	0	1	0	1	$\cdots$
$L(M_2)$	1	0	0	1	$\cdots$
$L(M_3)$	0	1	1	1	$\cdots$
$L(M_4)$	0	0	0	1	$\cdots$

Answer: $M_k \neq M_3$

$a^3 \notin L(M_k)$

$a^3 \in L(M_3)$

Similarly: $M_k \neq M_i$ for any i

Because either:

$$a^i \in L(M_k) \quad \text{and} \quad a^i \notin L(M_i)$$

or

$$a^i \notin L(M_k) \quad \text{and} \quad a^i \in L(M_i)$$

Therefore the machine M_k cannot exist

CONTRADICTION!!!

Therefore the language $\overline{L}$
is not recursively enumerable

End of proof

Observation:

There is no algorithm that describes $\overline{L}$

(otherwise it would be accepted by a Turing Machine)

A Language which is Recursively Enumerable and not Recursive

We want to find a language which

Is recursively enumerable

There is a Turing Machine that accepts the language

But not recursive

The machine doesn't necessarily halt on any input

We will prove that the language

$$L = \{a^i : a^i \in L(M_i)\}$$

Is recursively enumerable

but not recursive

Theorem:

The language $L = \{a^i : a^i \in L(M_i)$

is recursively enumerable

Proof:

We will give a Turing Machine that accepts L

Turing Machine that accepts L

For any input string w

- Write $w = a^i$
- Find Turing machine M_i

  (using the enumeration procedure for Turing Machines)
- Simulate M_i on input a^i
- If M_i accepts, then accept w

End of proof

Observation:

Recursively enumerable

$$L = \{a^i : a^i \in L(M_i)\}$$

Not recursively enumerable

$$\overline{L} = \{a^i : a^i \notin L(M_i)\}$$

(Thus, not recursive)

Theorem:

The language $L = \{a^i : a^i \in L(M_i)\}$

is not recursive

Proof:

Assume for contradiction that L is recursive

Then $\overline{L}$ is recursive:

Take the Turing Machine M that accepts L

M halts on any input

If M accepts then reject

If M rejects then accept

Therefore:

$\overline{L}$ recursive

But we know:

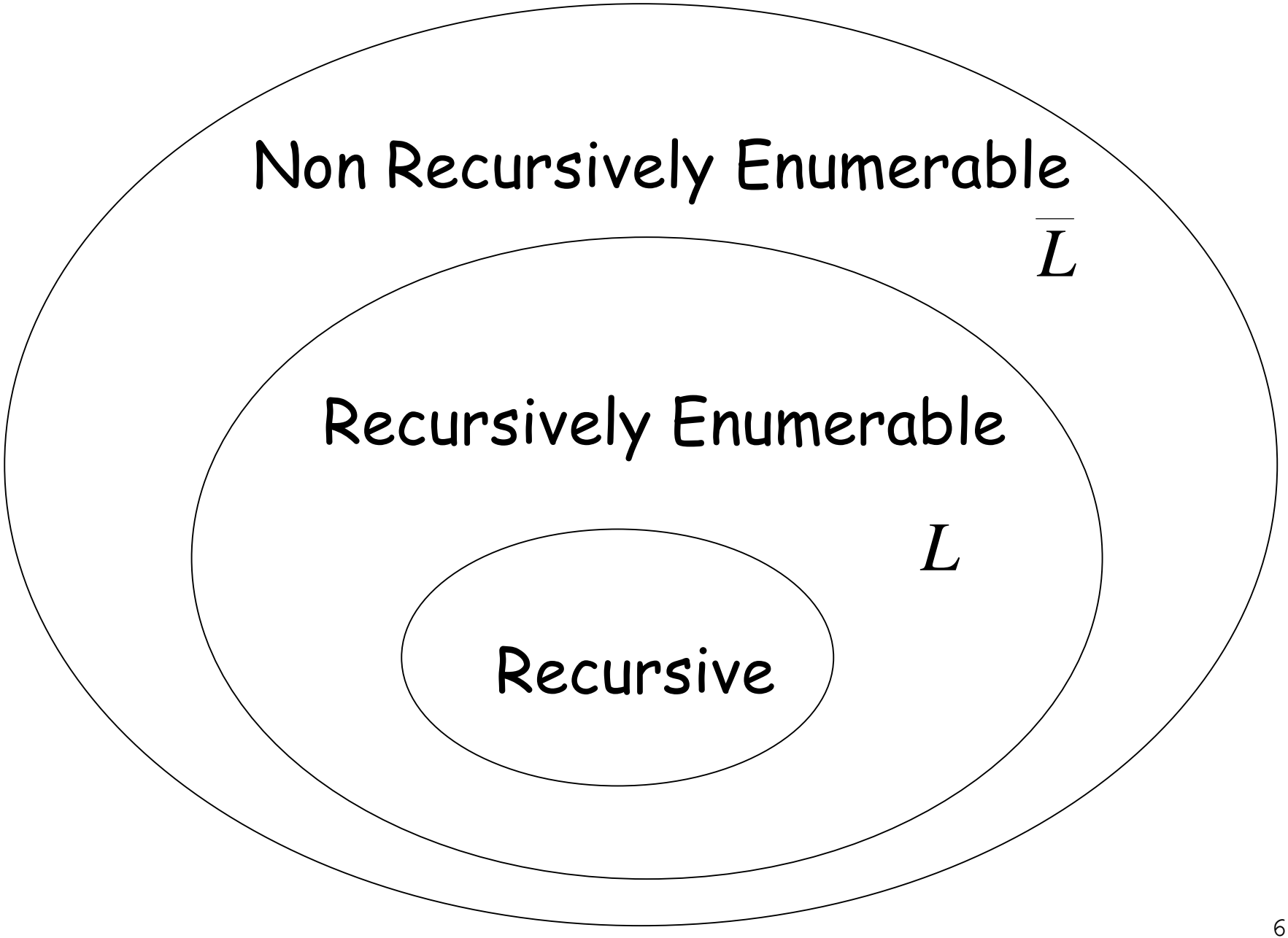
$\overline{L}$ not recursively enumerable

thus, not recursive

CONTRADICTION!!!!

Therefore, L is not recursive

End of proof


Non Recursively Enumerable
$\overline{L}$
Recursively Enumerable
L
Recursive