Decidable Languages

Recall that:

A language L is **Turing-Acceptable**
if there is a Turing machine M
that accepts L

Also known as: *Turing-Recognizable*
or
Recursively-enumerable
languages

For any string w :

$w \in L \Rightarrow M$ halts in an accept state

$w \notin L \Rightarrow M$ halts in a non-accept state

or loops forever

Definition:

A language L is **decidable**

if there is a Turing machine (decider) M

which accepts L

and halts on every input string

Also known as *recursive* languages

For any string w :

$w \in L \Rightarrow M$ halts in an accept state

$w \notin L \Rightarrow M$ halts in a non-accept state

Every decidable language is Turing-Acceptable

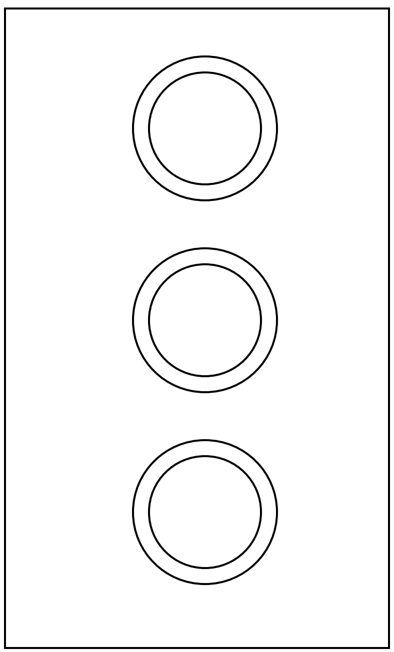
Sometimes, it is convenient to have Turing machines with single accept and reject states

These are the only halting states

That result to possible halting configurations

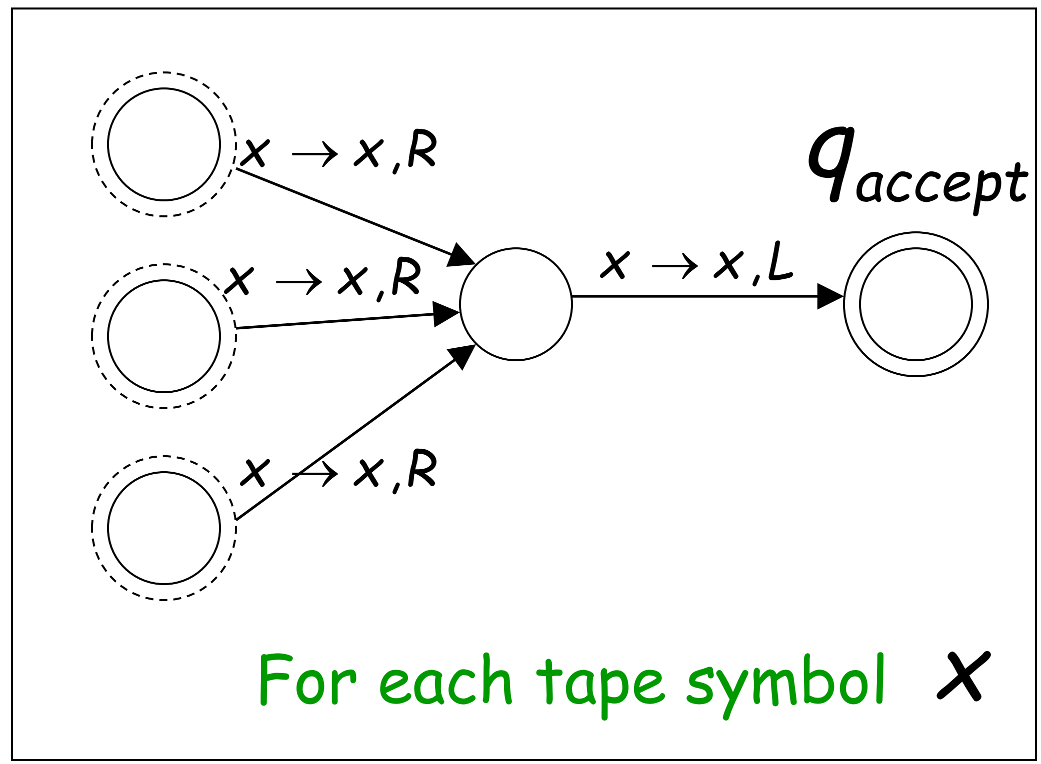
We can convert any Turing machine to have single accept and reject states

Old machine



Multiple accept states

New machine

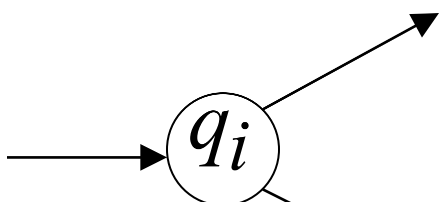


One accept state

Do the following for each possible halting state:

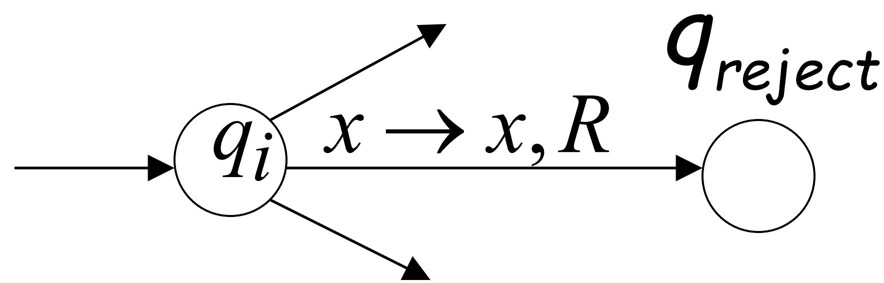
Old machine

For each



Multiple reject states

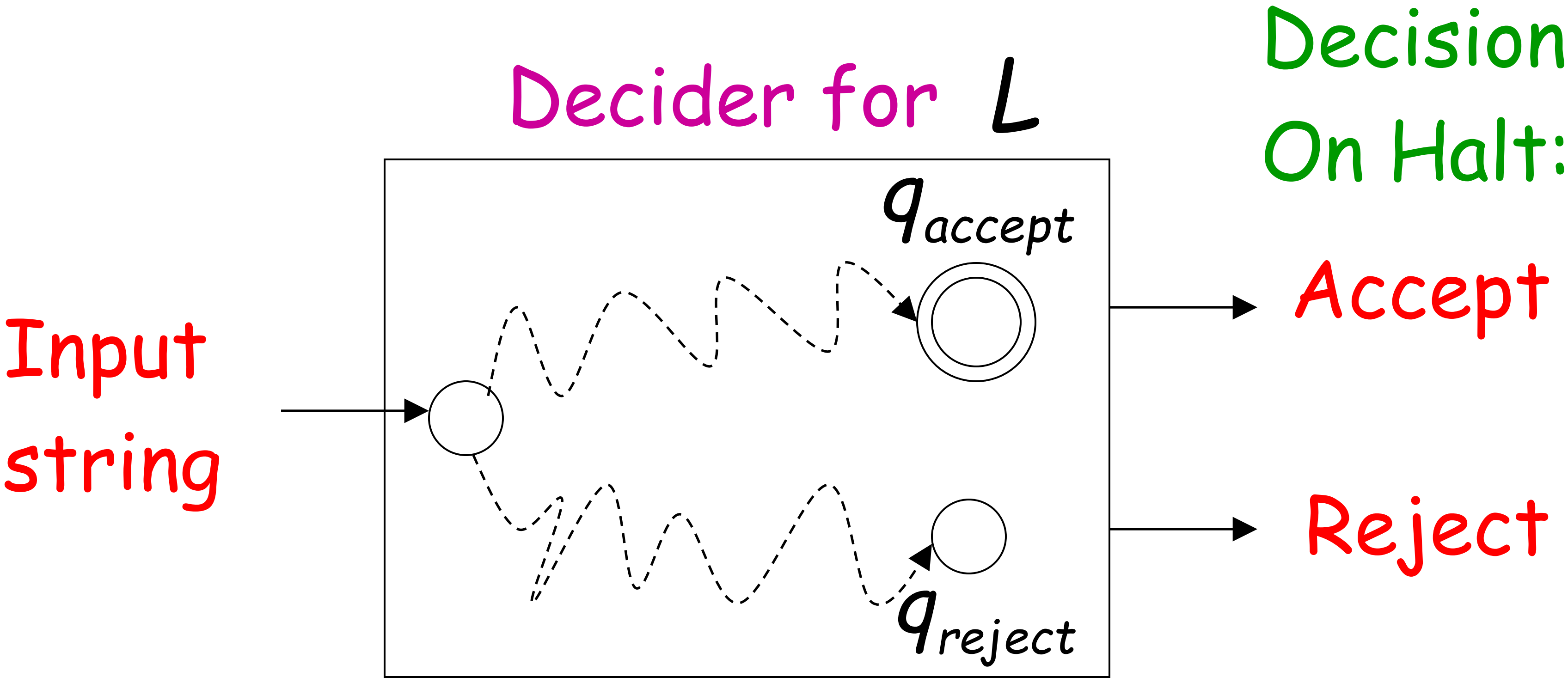
New machine



For all tape symbols x not used for read in the other transitions of q_i

One reject state

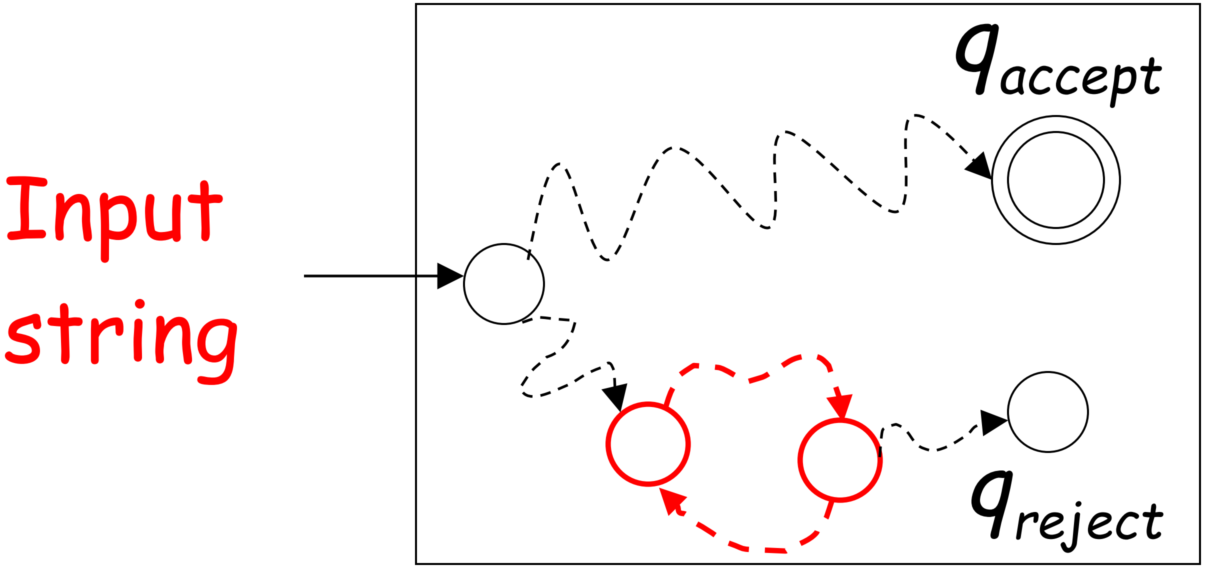
For a decidable language L :



For each input string, the computation halts in the accept or reject state

For a Turing-Acceptable language L :

Turing Machine for L



It is possible that for some input string the machine enters an infinite loop

Problem: Is number x prime?

Corresponding language:

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

We will show it is decidable

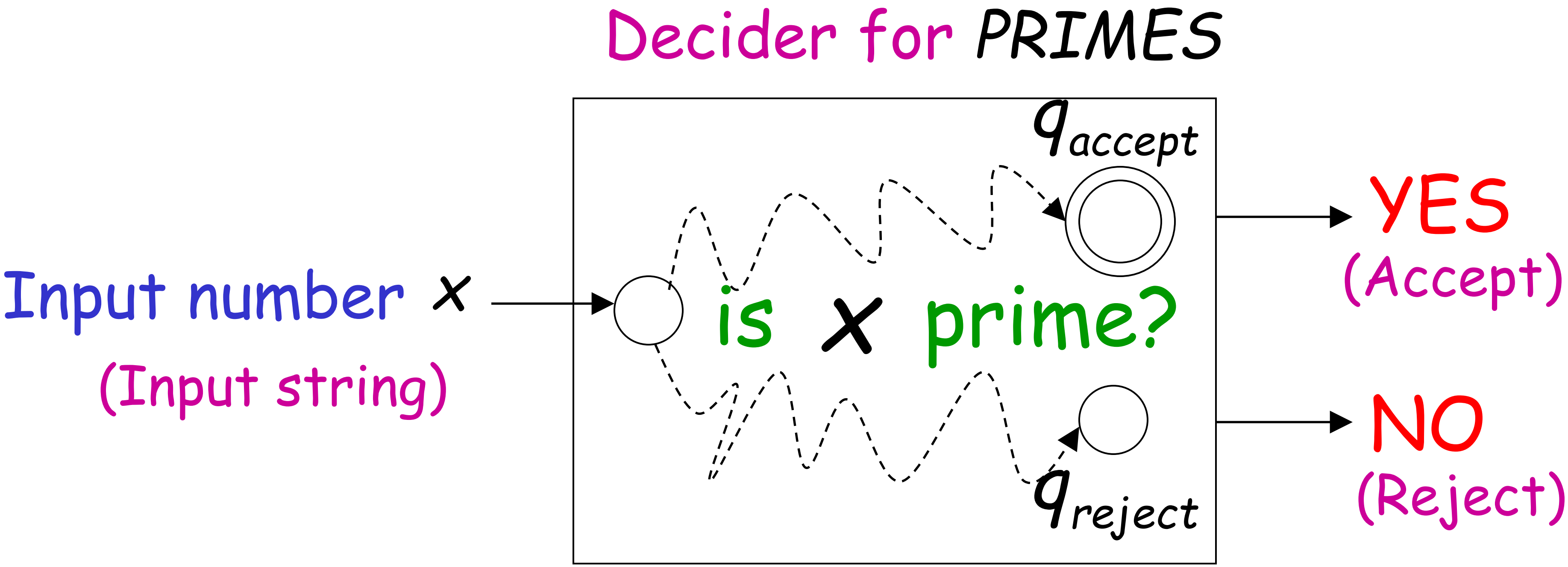
Decider for $PRIMES$:

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

the decider for the language
solves the corresponding problem



Theorem:

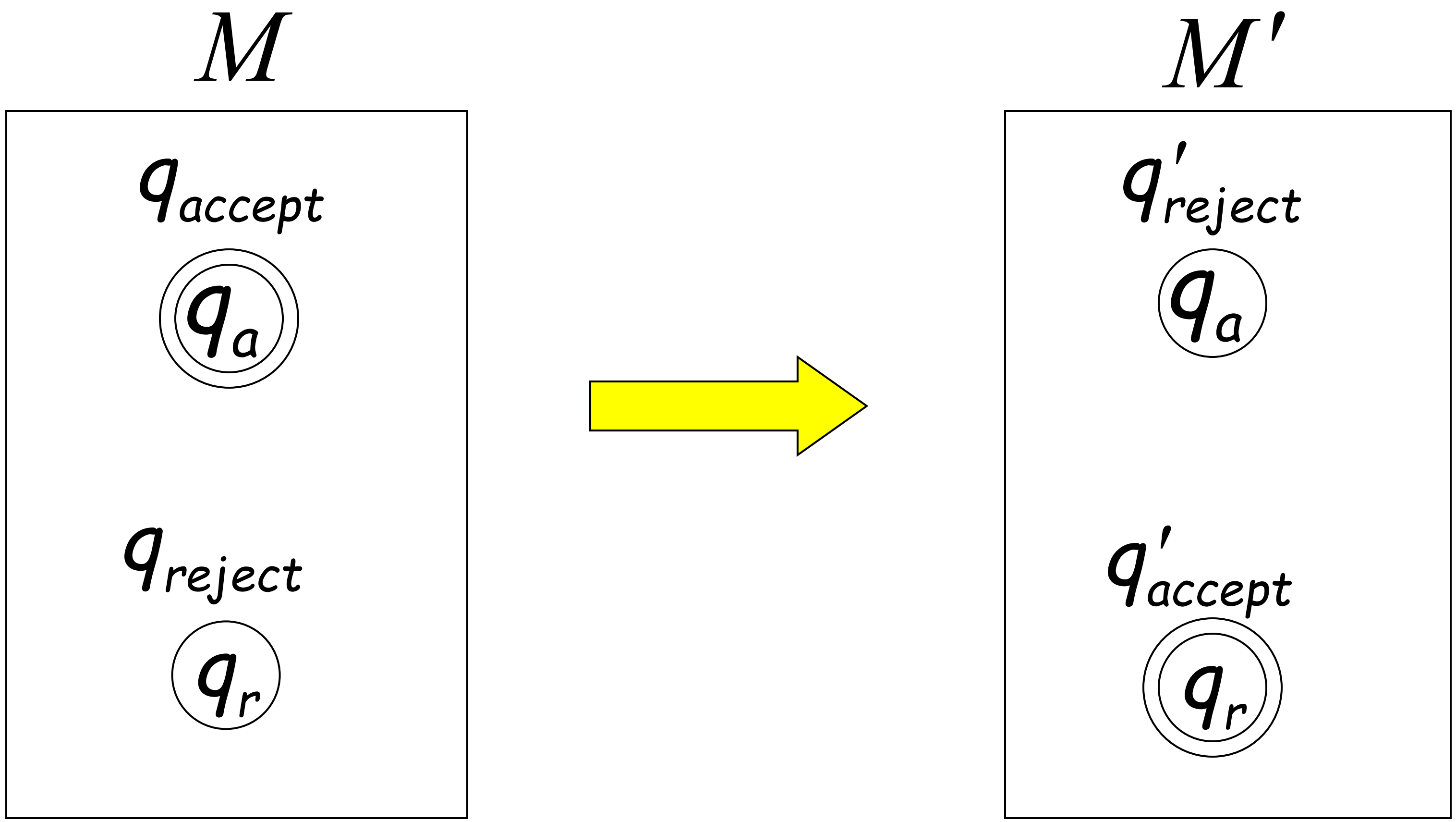
If a language L is decidable,
then its complement $\overline{L}$ is decidable too

Proof:

Build a Turing machine M' that
accepts $\overline{L}$ and halts on every input string

(M' is decider for $\overline{L}$)

Transform accept state to reject and vice-versa



Turing Machine M'

On each input string w do:

1. Let M be the decider for L

2. Run M with input string w

   If M accepts then reject

   If M rejects then accept

Accepts $\overline{L}$ and halts on every input string

END OF PROOF

Undecidable Languages

An undecidable language has no decider:

each Turing machine that accepts L

does not halt on some input string

We will show that:

There is a language which is

Turing-Acceptable and undecidable

We will prove that there is a language L:

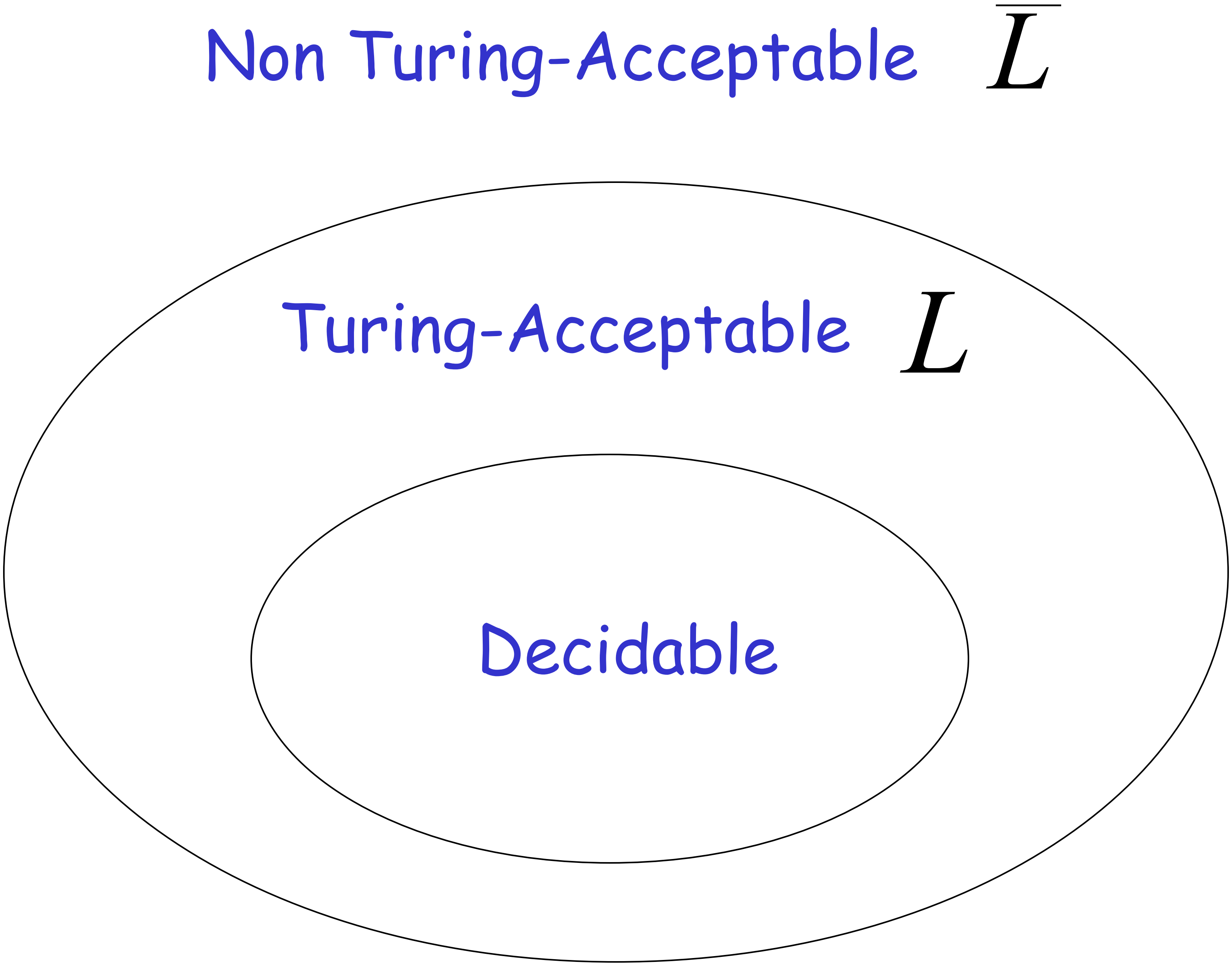
- $\overline{L}$ is **not** Turing-acceptable
  (not accepted by any Turing Machine)

- L is Turing-acceptable

the complement of a decidable language is decidable

$\Downarrow$

Therefore, L is undecidable


Non Turing-Acceptable $\overline{L}$
Turing-Acceptable L
Decidable

A Language which is not Turing Acceptable

Consider alphabet $\{a\}$

Strings of $\{a\}^+$:

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

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

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

They are countable:

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

(There is an enumerator that generates them)

Each machine accepts some language over $\{a\}$

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

$$L(M_1),\ L(M_2),\ L(M_3),\ L(M_4),\ \ldots$$

Note that it is possible to have

$$L(M_i) = L(M_j) \quad \text{for} \quad i \neq j$$

Since, a language could be accepted by more than one Turing machine

Example language accepted by M_i

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

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

Binary 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	$\cdots$

Example of binary representations

	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 of 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 $\overline{L}$

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

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

$\overline{L}$ consists of 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 Turing-Acceptable

Proof:

Assume for contradiction that

$\overline{L}$ is Turing-Acceptable

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: $M_k = M_1$? $\qquad L(M_k) = \overline{L}$

	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	$\cdots$

$$a^1 \in L(M_k)$$

Answer: $M_k \neq M_1$

$$a^1 \notin L(M_1)$$

	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: $M_k = M_2$?

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

	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	$\cdots$

$$a^2 \in L(M_k)$$

Answer: $M_k \neq M_2$

$$a^2 \notin L(M_2)$$

	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: $M_k = M_3$?

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

	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	$\cdots$

$$a^3 \notin L(M_k)$$

Answer: $M_k \neq M_3$

$$a^3 \in L(M_3)$$

Similarly: $M_k \neq M_i$ for any i

Because either:

$a^i \in L(M_k)$ or $a^i \notin L(M_k)$

$a^i \notin L(M_i)$ $a^i \in L(M_i)$

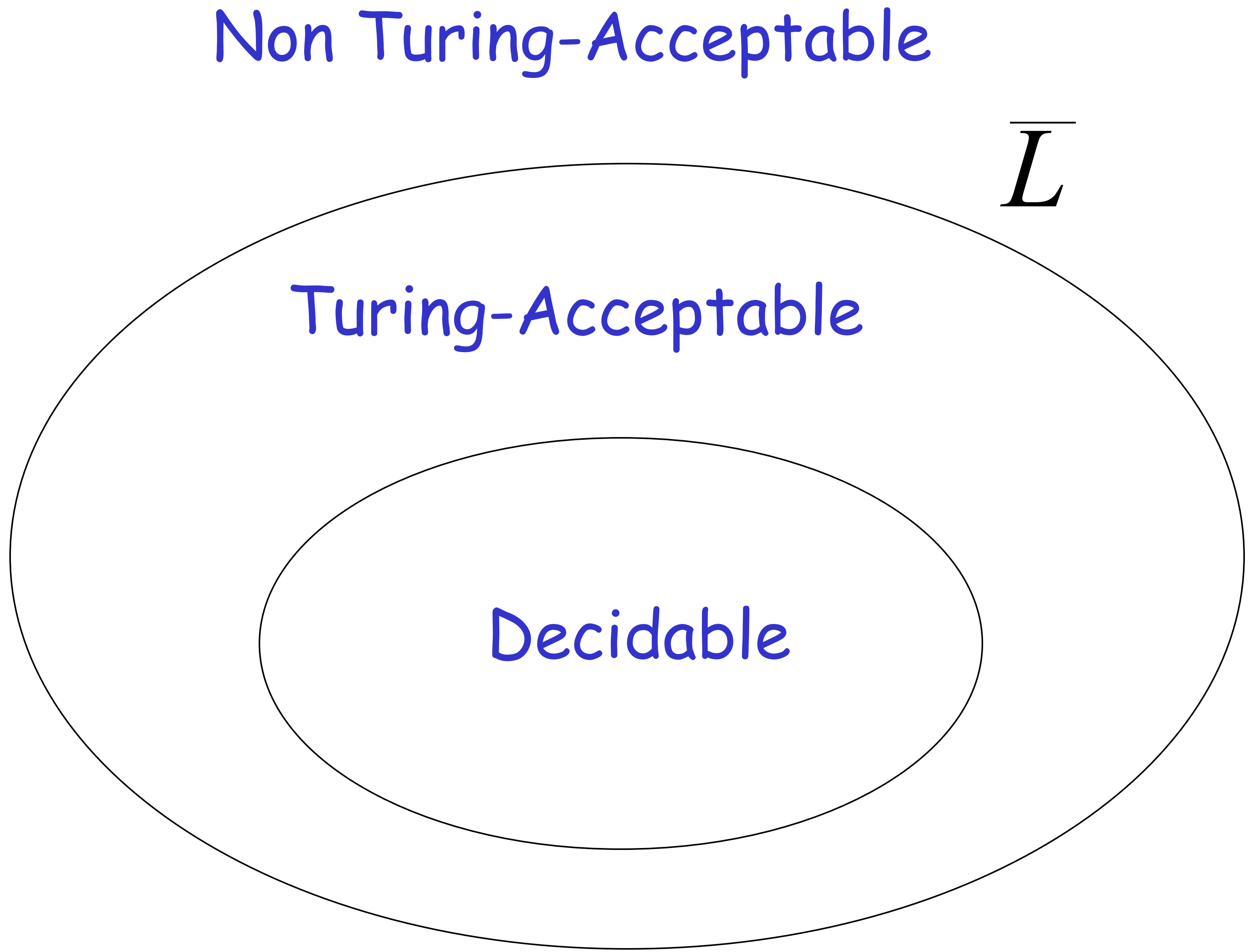
$\Downarrow$

the machine M_k cannot exist

$\Downarrow$

$\overline{L}$ is not Turing-Acceptable

End of Proof


Non Turing-Acceptable
$\overline{L}$
Turing-Acceptable
Decidable

A Language which is Turing-Acceptable and Undecidable

We will prove that the language

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

Is Turing-Acceptable

Undecidable

There is a Turing machine that accepts L

Each machine that accepts L doesn't halt on some input string

	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	$\cdots$

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

Theorem: The language

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

Is Turing-Acceptable

Proof: We will give a Turing Machine that accepts L

Turing Machine that accepts L

For any input string w

- Compute i, for which $w = a^i$

- Find Turing machine M_i

  (using the enumerator for Turing Machines)

- Simulate M_i on input a^i

- If M_i accepts, then accept w

End of Proof

Observation:

Turing-Acceptable

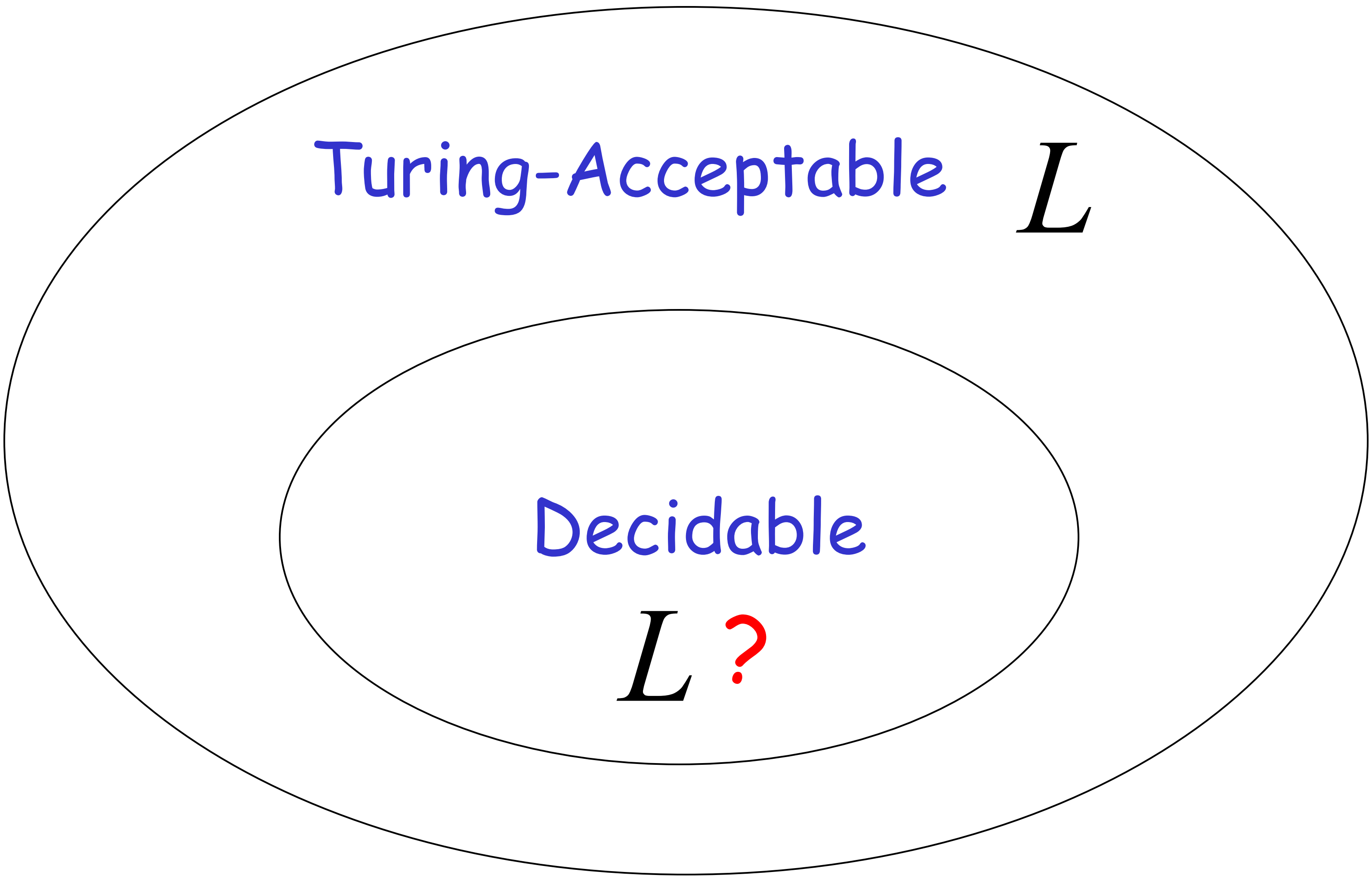
$$L = \{a^i : a^i \in L(M_i)\}$$

Not Turing-acceptable

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

(Thus, $\overline{L}$ is undecidable)

Non Turing-Acceptable $\overline{L}$



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

is undecidable

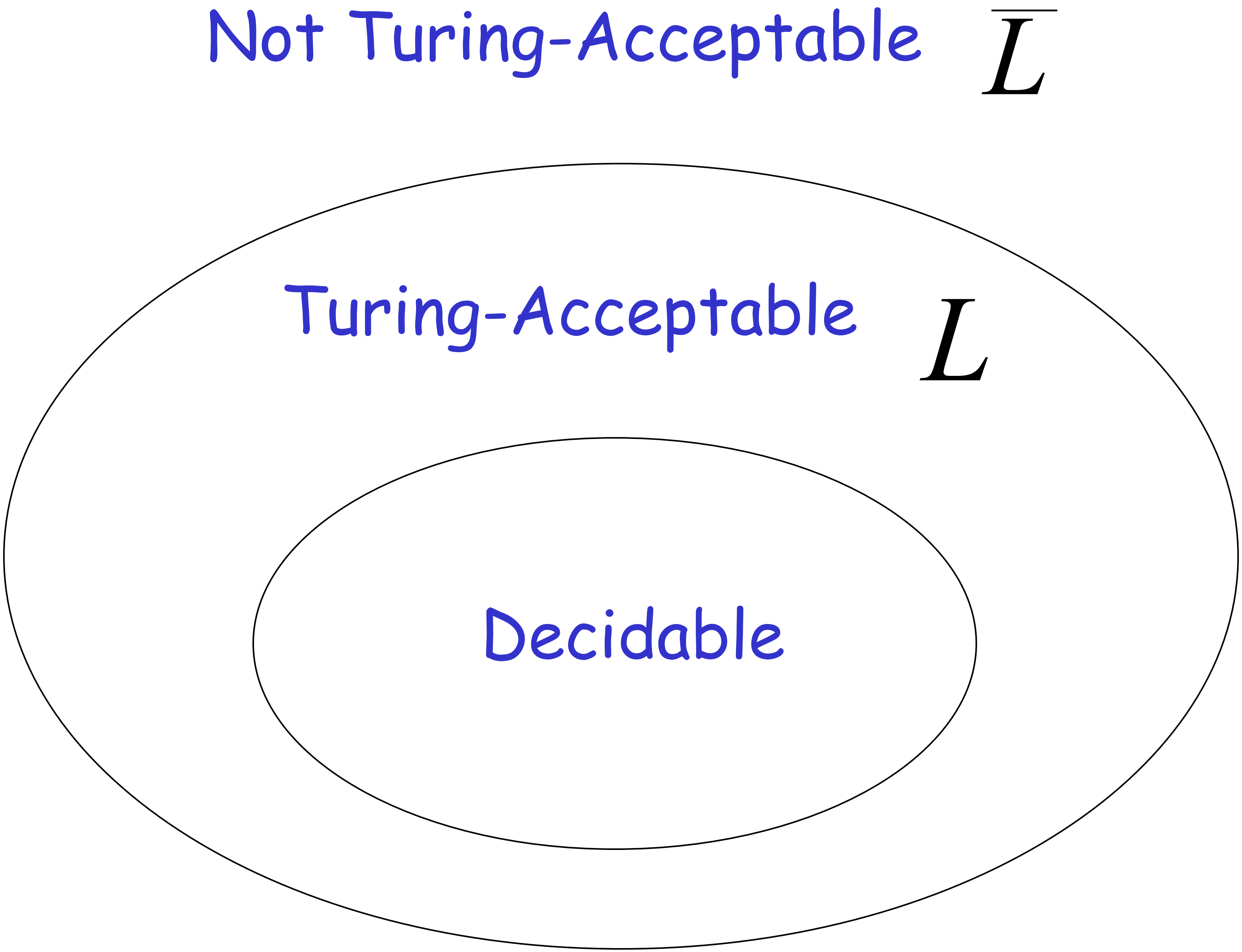
Proof: If L is decidable

$\Downarrow$ the complement of a decidable language is decidable

Then $\overline{L}$ is decidable

However, $\overline{L}$ is not Turing-Acceptable!

Contradiction!!!!


Not Turing-Acceptable $\overline{L}$
Turing-Acceptable L
Decidable