SASTRA
ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION
DEEMED TO BE UNIVERSITY
(U/S 3 OF THE UGC ACT, 1956)
THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

CSE211-Formal Languages and Automata Theory

U4L4_Recursively Enumerable and not Recursive

Dr. P. Saravanan
School of Computing
SASTRA Deemed University

Agenda

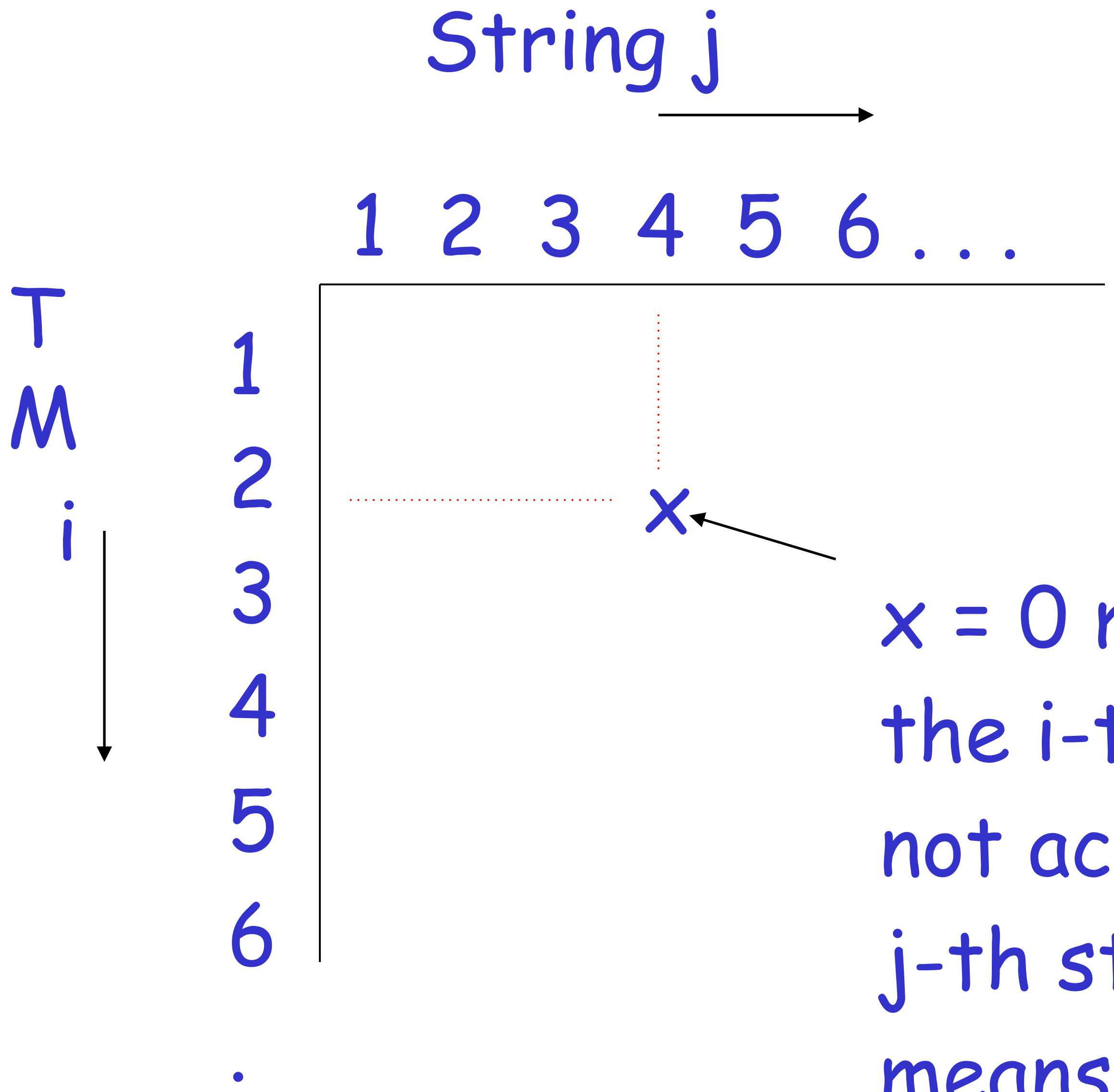
- Recap:
  - Diagonalization
  - Diagonalization language
  - A Language which is not Recursively Enumerable- proof
- A language which is Recursively Enumerable and not Recursive

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

Table of Acceptance



x = 0 means the i-th TM does not accept the j-th string; 1 means it does.

Diagonalization Again

- Whenever we have a table like the one on the previous slide, we can *diagonalize* it.
  - That is, construct a sequence D by complementing each bit along the major diagonal.
- Formally, $D = a_1a_2\ldots$, where $a_i = 0$ if the (i, i) table entry is 1, and vice-versa.

Diagonalization - (2)

Consider the diagonalization language

L_d = {w | w is the i-th string, and the i-th TM does not accept w}.

Consider alphabet $\{a\}$

Strings: $a, \; aa, \; aaa, \; aaaa, \; \sqcup$

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

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

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 of the 0's in the diagonal

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

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

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	$\cdots$
$L(M_1)$	$\boxed{0}$	1	0	1	$\cdots$
$L(M_2)$	1	$\boxed{0}$	0	1	$\cdots$
$L(M_3)$	0	1	1	1	$\cdots$
$L(M_4)$	0	0	0	1	$\cdots$

Question: $M_k = M_1$?

$$\begin{array}{c|ccccc}
 & a^1 & a^2 & a^3 & a^4 & \cdots \\ \hline
L(M_1) & \boxed{0} & 1 & 0 & 1 & \cdots \\
L(M_2) & 1 & \boxed{0} & 0 & 1 & \cdots \\
L(M_3) & 0 & 1 & 1 & 1 & \cdots \\
L(M_4) & 0 & 0 & 0 & 1 & \cdots
\end{array}$$

Answer: $M_k \neq M_1$

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

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

$$\begin{array}{c|ccccc}
 & a^1 & a^2 & a^3 & a^4 & '' \\
\hline
L(M_1) & \textcircled{0} & 1 & 0 & 1 & '' \\
L(M_2) & 1 & \textcircled{0} & 0 & 1 & '' \\
L(M_3) & 0 & 1 & 1 & 1 & '' \\
L(M_4) & 0 & 0 & 0 & 1 & ''
\end{array}$$

Question: $M_k = M_3$?

	a^1	a^2	a^3	a^4	$\cdots$
$L(M_1)$	$\boxed{0}$	1	0	1	$\cdots$
$L(M_2)$	1	$\boxed{0}$	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:

$a^i \in L(M_k)$

$a^i \notin L(M_i)$

or

$a^i \notin L(M_k)$

$a^i \in L(M_i)$

Therefore, the machine M_k cannot exist

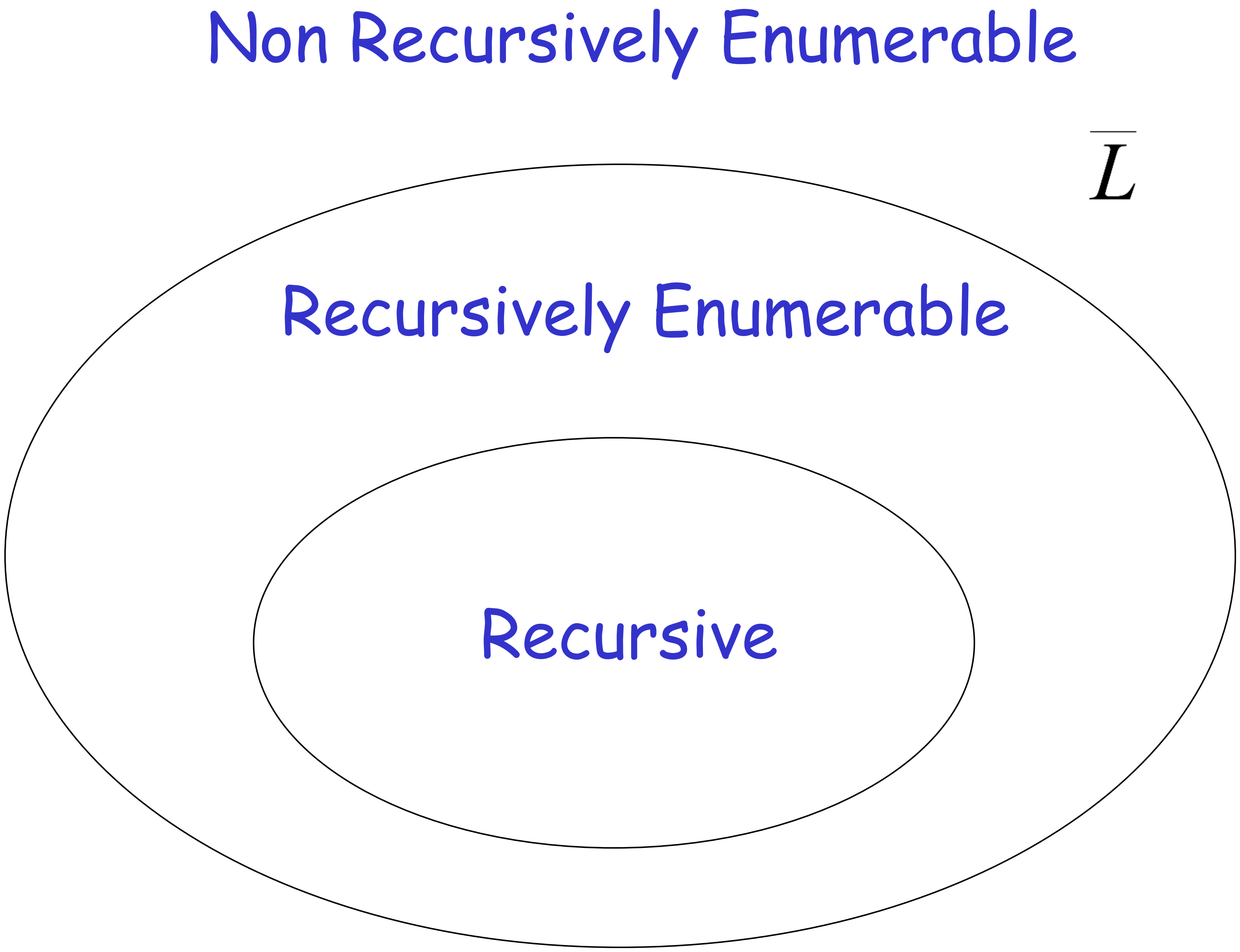
Therefore, the language $\overline{L}$ is not recursively enumerable

End of Proof

Observation:

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

(otherwise $\overline{L}$ would be accepted by some Turing Machine)


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

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

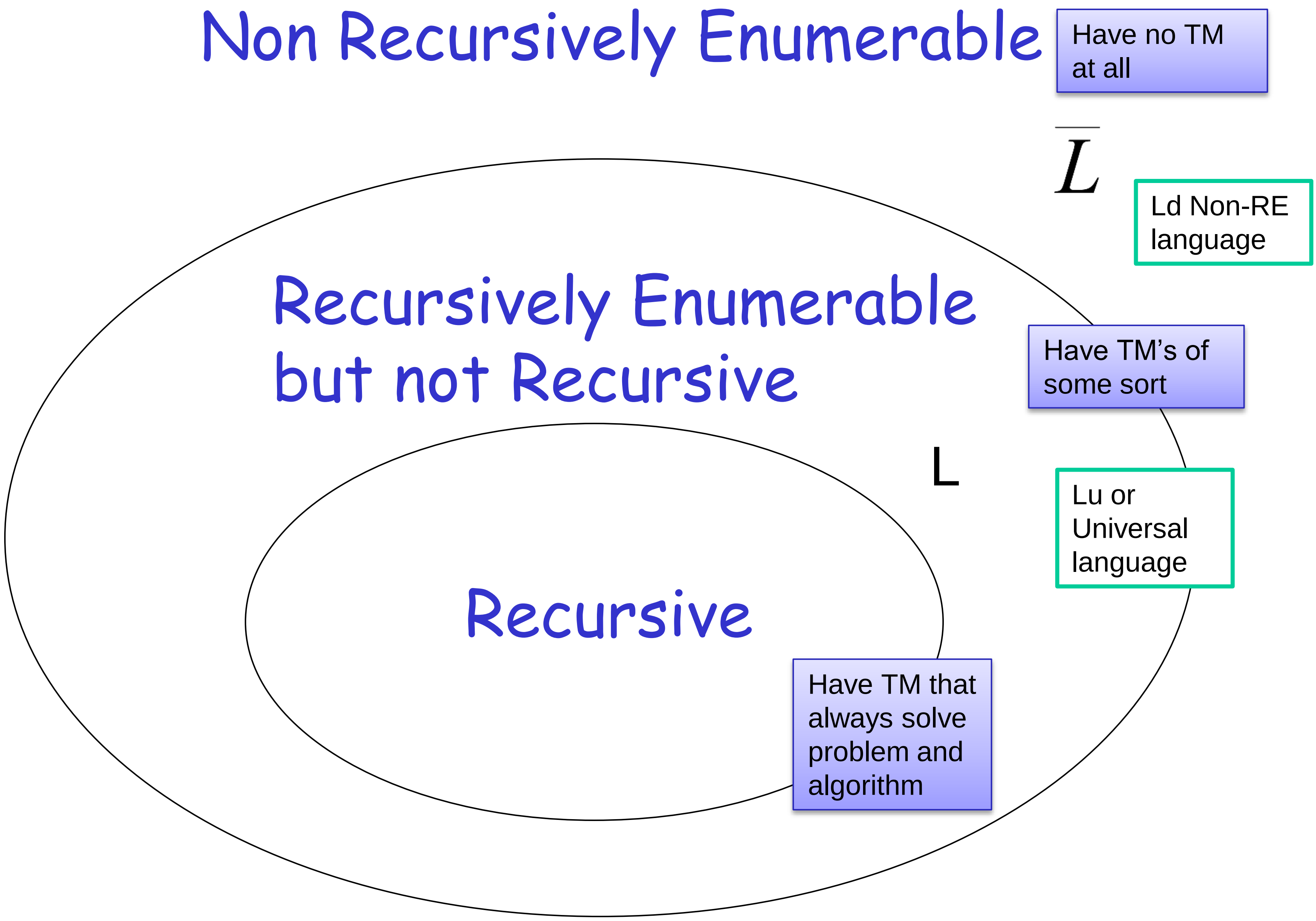
The machine doesn't halt on some input

Recursive language

- A TM of this type corresponds to our informal notion of an 'algorithm'
- a well defined sequence of steps that always finishes and produces an answer
- If we think of the language L as a "problem" as will be the case frequently then problem L is called
  - decidable if it is a recursive language and
  - undecidable if it is not recursive language

Why recursive

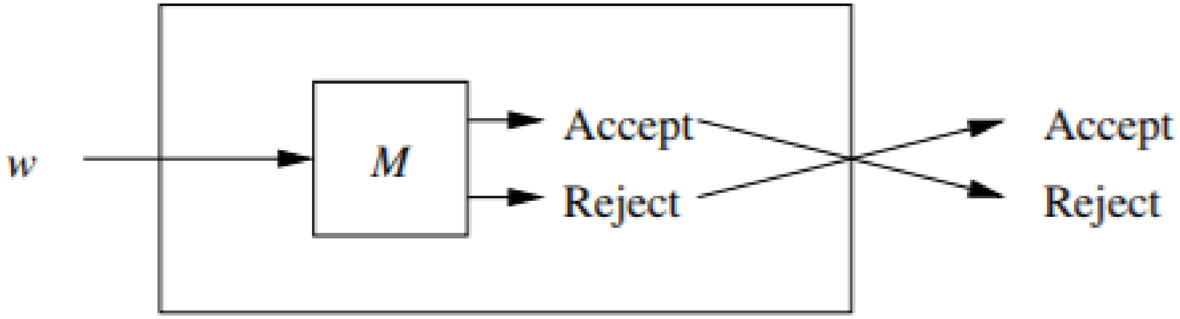
- The existence or nonexistence of an algorithm to solve a problem is often of more importance than the existence of TM to solve the problem.
- The Turing machines that are not guaranteed to halt may not give us enough information ever to conclude that a string is not in the language
- so there is a sense in which they have not solved the problem


Non Recursively Enumerable
Have no TM at all
$\overline{L}$
Ld Non-RE language
Recursively Enumerable but not Recursive
Have TM's of some sort
L
Lu or Universal language
Recursive
Have TM that always solve problem and algorithm

Theorem:

If L is a recursive so $\overline{L}$ is recursive language

TM for $\overline{L}$



We will prove that the language

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

Is recursively enumerable
but not recursive

$$
\begin{array}{c|ccccc}
 & a^1 & a^2 & a^3 & a^4 & \cdots \\
\hline
L(M_1) & 0 & 1 & 0 & 1 & \cdots \\
L(M_2) & 1 & 0 & 0 & 1 & \cdots \\
L(M_3) & 0 & 1 & \enclose{circle}{1} & 1 & \cdots \\
L(M_4) & 0 & 0 & 0 & \enclose{circle}{1} & \cdots
\end{array}
$$

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

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

- Compute i , for which $w = a^i$
- Find Turing machine M_i

  (using an 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, also 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:

$\overline{L}$ is recursive

But we know:

$\overline{L}$ is not recursively enumerable

thus, not recursive

CONTRADICTION!!!!

Therefore, L is not recursive

End of Proof

References

John E. Hopcroft, Rajeev Motwani and Jeffrey D. Ullman, *Introduction to Automata Theory*, Languages, and Computation, Pearson, 3rd Edition, 2011.

Peter Linz, *An* Introduction to Formal Languages and *Automata*, Jones and Bartle Learning International, United Kingdom, 6th Edition, 2016.

Next Class: Unit IV

Universal Language

Thank you.