Department of Computer & Information Sciences
Pakistan Institute of Engineering and Applied Sciences

Recursively Enumerable and Recursive Languages

Based on slides by Costas Busch [http://csc.lsu.edu/~busch] Umar Faiz

http://www.pieas.edu.pk/umarfaiz/cis317

1

Recursive Languages

Definition:

A recursive language L is a formal language for which there exists a TM that will halt and accept an input string in L, and halt and reject, otherwise.

Example:

The decision problem of determining whether a natural number is a perfect square (represented by using the string an) is decidable.

2

Recursive Languages

Let L be a recursive language and M 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

Costas Busch - RPI 3

Recursively Enumerable Languages

Definition:

A language is recursively enumerable if some Turing machine accepts it

Let L be a recursively enumerable language and M 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 and loops forever

Costas Busch - RPI 4

Non Recursively Enumerable

We will prove:

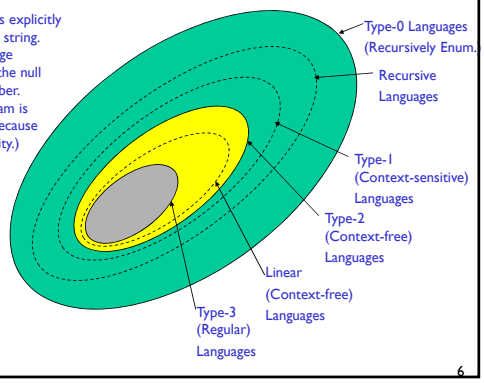
1. There is a specific language which is not recursively enumerable (not accepted by any Turing Machine)

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

Costas Busch - RPI 5

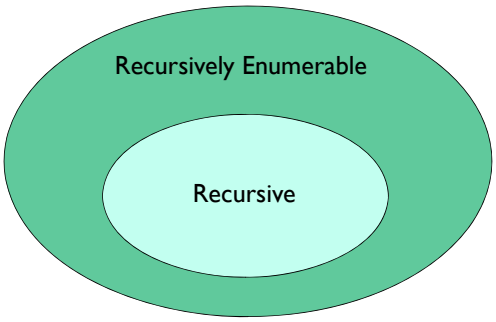
Context-Free Languages

Technical Note:
Type-1 languages explicitly exclude the null string. All other language families permit the null string as a member. (The Venn diagram is in slight error because of this technicality.)



6

Non Recursively Enumerable



A Language which
is not
Recursively Enumerable

Non Recursively Enumerable

- We want to find a language that is Non Recursively Enumerable
- This language is not accepted by any Turing Machine

Non Recursively Enumerable

Consider alphabet $\{a\}$

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

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

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

They are countable:

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

Non Recursively Enumerable

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	$\cdots$

Non Recursively Enumerable

	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	$\cdots$

Non Recursively Enumerable

Consider the language

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

L consists from the 1's in the diagonal

Non Recursively Enumerable

	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	$\cdots$

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

Non Recursively Enumerable

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

Non Recursively Enumerable

	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	$\cdots$

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

Non Recursively Enumerable

Theorem:

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

Non 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: $M_k \neq M_3$

$a^3 \notin L(M_k)$

$a^3 \in L(M_3)$

Non Recursively Enumerable

Similarly: $M_k \neq M_i$ for any i

Because either:

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

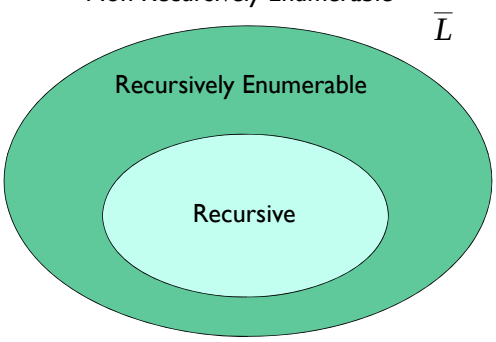
Non Recursively Enumerable

Therefore, the machine M_k cannot exist. Therefore, the language $\overline{L}$ is not recursively enumerable

Observation:

There is no algorithm that describes $\overline{L}$ (otherwise would be accepted by some Turing Machine)

Non Recursively Enumerable



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

We will prove that the language

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

Is recursively enumerable but not recursive

	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 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

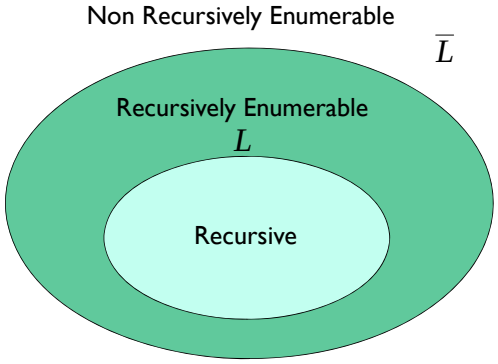
But we know:

$\overline{L}$ is not recursively enumerable

thus, not recursive

CONTRADICTION!!!!

Therefore, L is not recursive



Turing acceptable languages
and
Enumeration Procedures

We will prove:

(weak result)

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

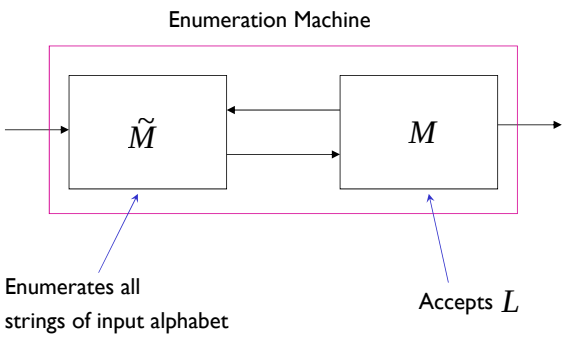
(strong result)

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:



If the alphabet is $\{a,b\}$ then $\tilde{M}$ can enumerate strings as follows:

a
b
aa
ab
ba
bb
aaa
aab
.....

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

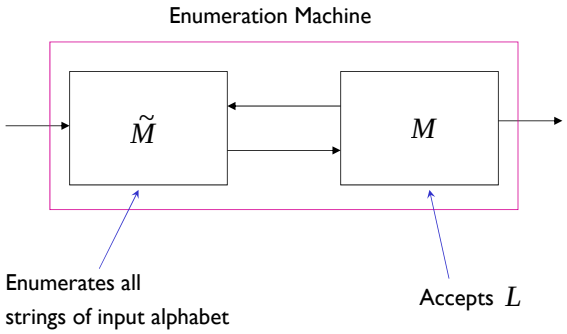
Example: $L = \{b, ab, bb, aaa, ...\}$

$\tilde{M}$	$L(M)$	Enumeration Output
a		
b	b	b
aa		
ab	ab	ab
ba		
bb	bb	bb
aaa	aaa	aaa
aab		
.....	.....	.....

Theorem:

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

Proof:



If the alphabet is $\{a,b\}$ then $\tilde{M}$ can enumerate strings as follows:

a
b
aa
ab
ba
bb
aaa
aab

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

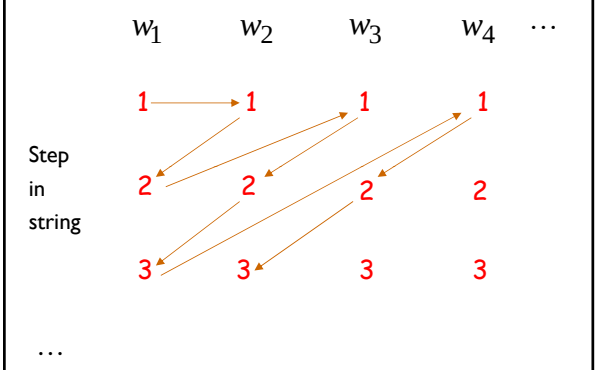
$\tilde{M}$ Generates third string w_3

M executes first step on w_3

second step on w_2

third step on w_1

And so on.....



If for any string w_i machine halts in a final state then it prints M on the output w_i

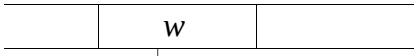
End of Proof

Theorem:

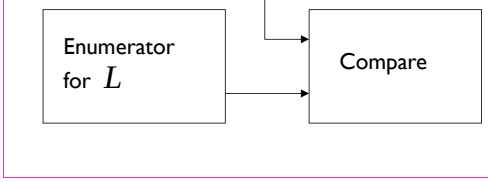
If for language L there is an enumeration procedure then L is recursively enumerable

Proof:

Input Tape



Machine that accepts L



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

We have proven:

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

Context-Free Languages

Technical Note:
Type-1 languages explicitly exclude the null string. All other language families permit the null string as a member. (The Venn diagram is in slight error because of this technicality.)