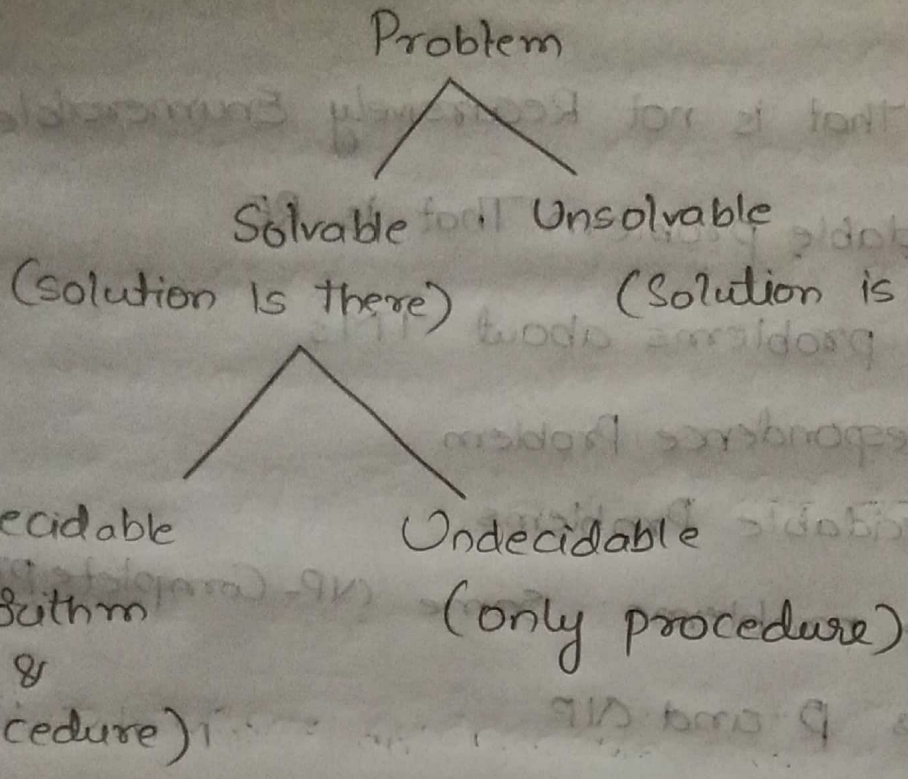
=> Any problem can be solvable (or) unsolvable.



Decidability:-

A problem is said to be decidable if we can construct a turing machine which will in finite amount of time for every inputs and give answers Yes (or) no

A decidable problem can be easily solved with the help of an algorithmic approach ie. by computer.

Example:

1. Sun rises in the east ? Yes
2. Earth moves around the sun : Yes

Examples:-

i. Equivalence of 2 Regular lang:

Given 2 Regular lang's, there is an algorithm and TM to decide whether two R.L's are equal & not

ii. Finiteness of Regular lang: Given a RL, there is an algorithm & TM to decide whether R.L is finite & not.

iii. Emptiness of CFL: Given a CFL, there is an alg whether CFL is empty (or) not.

Decidable problems for FA:

a. For a given FA "M" and input alphabet Σ

1. $L(M) = \phi$ is decidable
2. $w \in L(M)$, where w is a string in Σ is decidable
3. $L(M) = \Sigma^*$ is decidable.

b. For given two FA's M_1 and M_2 and input alphabet

$L(M_1) = L(M_2)$ is decidable.

Decidable problems for Regular languages:-

The following operations one two regular Languages L_1 and L_2 are decidable.

① $L_1 \cup L_2$ ② $L_1 \cap L_2$ ③ $L_1 \cdot L_2$ ④ L_1^* (or) L_2

⑤ L_1^* (or) L_2^* | ⑥ L_1^R (or) L_2^R. Moreover

⑦ $L_1 \subset L_2$ is decidable

Decidable problems for CFGs and CFLs :-

1. Two CFL's L_1 and L_2 are decidable if $L_1 \cup L_2$ is CFL

2. Two CFL's L_1 and L_2 are decidable if $L_1 L_2$ is CFL.

3. A CFL L, decidable if L^* is CFL.

4. A regular language L_1, and a CFL L_2 is decidable if $L_1 \cup L_2$ is CFL.

5. A regular lang L_1 and a CFL L_2 is decidable if $L_1 \cap L_2$ is CFL.

6. For a given CFG 'G'

   i. $L(G) = \emptyset$ is decidable

   ii. $L(G)$ is finite (or) infinite is decidable

Decidable Problems for Recursive and Recursive Enumerable Languages :-

A recursive enumerable language is decidable if it is recursive.

1. For a given recursive enumerable language L, L' is recursive.

2. For given two recursive languages L_1 and L_2

i. $L_1 \cup L_2$ is recursive

ii. L_1' is recursive (or)

L_2' is recursive.

Undecidability:-

A problem is said to be undecidable if there is no turing machine, which will always halt in finite amount of time to give answers as Yes (or) No.

A problem which cannot be solved by any algorithmic approach is referred as undecidable.

A given problem is undecidable if,

* It's corresponding language is non-recursive

and

* It has no solution (or) answer (or) algorithm

Examples:-

1. Ambiguity of CFL: Given a CFL, there is no TM which will always halt in finite amount of time and given ans whether the lang is ambiguous or not

2. Equivalence of 2 CFL's: Given 2 CFL, there is no TM which will always halt in finite amount of time and give answer whether 2 CFL's are equal (or) not.

3. Everything (or) Completeness of CFG: Given a CFG and input alphablet, whether CFG will generate all possible strings of input alphabet Σ^* is undecida

Note:

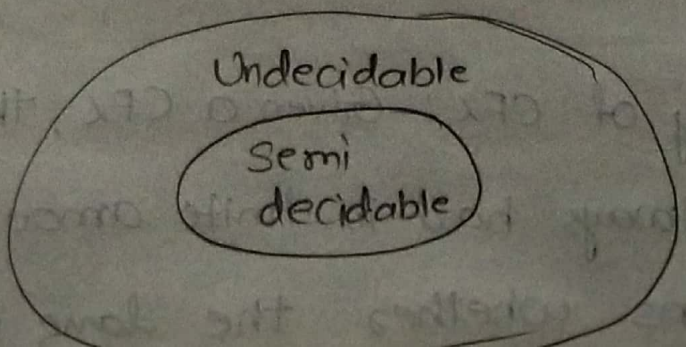
Two popular undecidable problems are

Halting problem of TM and

PCP [Post Correspondence Problem]

Semi decidable problem:

A semi decidable problem is subset of undecidable problem for which TM will always ha in finite amount of time for answer as Yes an may (or) may not halt for ans as NO.

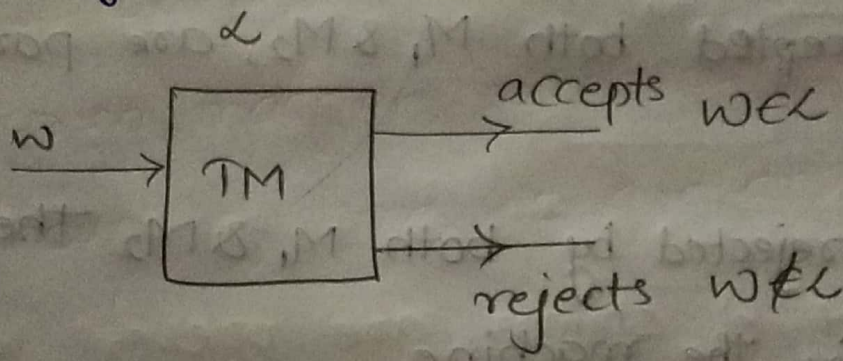


Recursive Language:

A Language "L" is said to be recursive if there exists a TM, which will accepts the all the string which are in "L" and rejects the strings which are not in "L"

⇒ Recursive Language is Decidable language.

⇒ This type of turing machine is called as Halting Turing Machine.

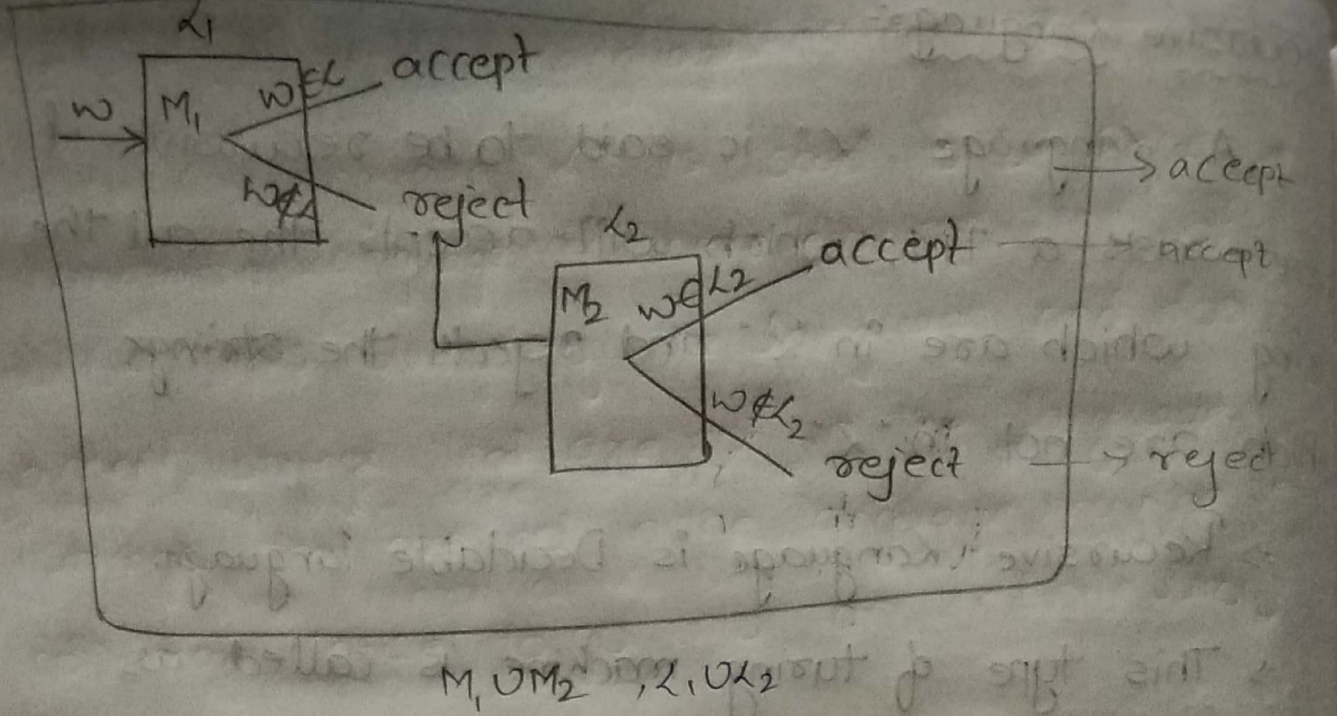


Properties of Recursive language:

Recursive language is closed under union, intersection and Complement.

Union: L_1, L_2 are two recursive languages, then union of L_1 & L_2 is also recursive.

$L_1 \cup L_2 \Rightarrow$ recursive.



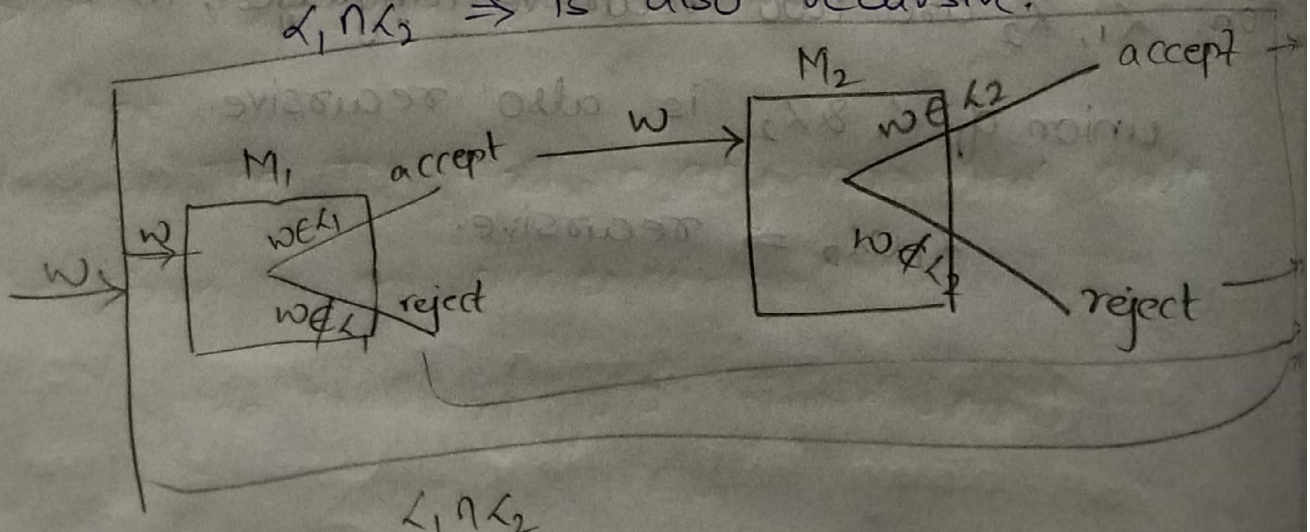
⇒ Overall, the Combine machine accepts the string which are accepted both M_1 & M_2, are part of the L_1 & L_2

⇒ Input is rejected by both M_1 & M_2 then it is rejected by the machine.

Intersection:

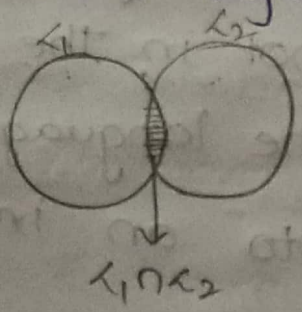
L_1 & L_2 are two recursive language, then the intersection of L_1 & L_2 is also recursive

$L_1 \cap L_2$ ⇒ is also recursive.



The ~~language~~ string which accepted by M_1 only then only string is accepted by M_2.

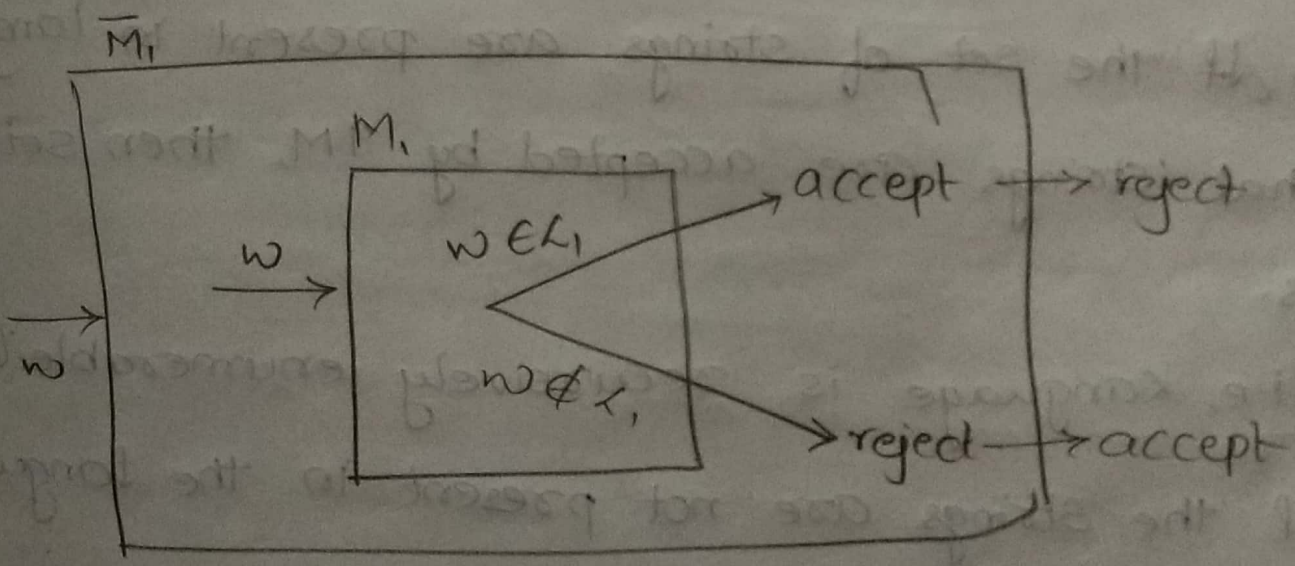
The string which is rejected by M_1, M_2 then the overall machine also rejects the language.



Complement:

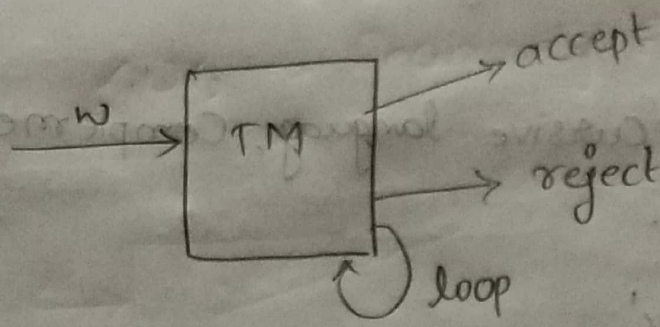
L_1 is a recursive language, complement of L_1 is also recursive.

$L_1 \rightarrow L_1'$



Recursive Enumerable Language:-

A Language "L" is said to be Recursive enumerable language if there exists a TM which will accepts strings of the language, and does not accept strings that are not in the language. [strings that are not in the language may be reject (or) may cause to go into an infinite loop].



⇒ If the set of strings are present in langua all that strings are accepted by TM, then set of strings.

i.e, Language is recursively enumerable lang

⇒ If the strings are not present in the language then

1, The TM Halt in any non-final state and the string is not accepted by TM.

⇒ the TM may goto infinite loop

⇒ To overcome the problem of infinite loop in recursive enumerable lang, restrict the TM in such away that the string is present at the lang, then the string is accepted and halts at final state.

⇒ If the string is not present in the language, then then string is rejected and halts at non-final state (but not going to infinite loop)

Such a TM is called as Halting TM

⇒ Every recursive language is recursive enumerable language.

⇒ But every recursive enumerable language is not recursive language.

Recursive enumerable language is Partially decidable language.

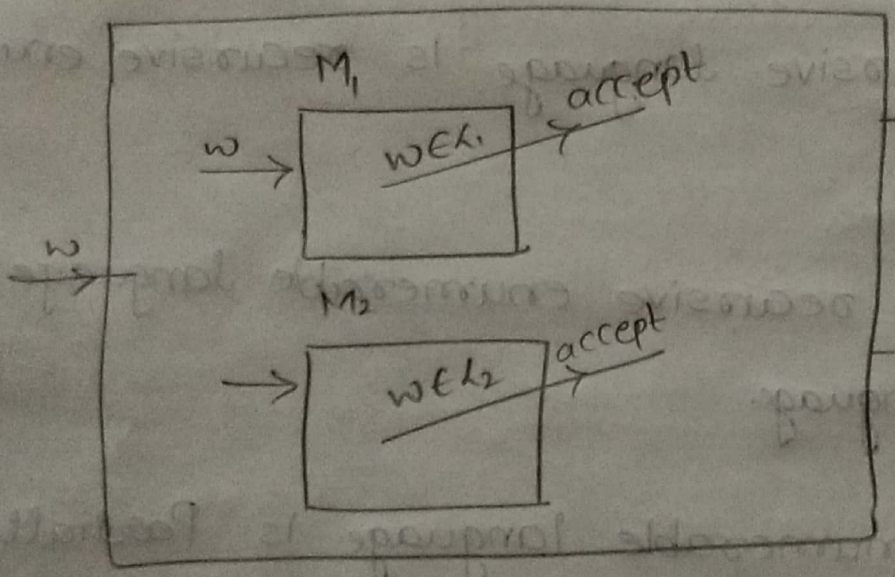
Properties of Recursively Enumerable language:

Union:

Recursively enumerable language is closed under union, intersection.

Union: L_1 & L_2 are recursively enumerable lang, then union of L_1 & L_2 is also recursively enumerable language

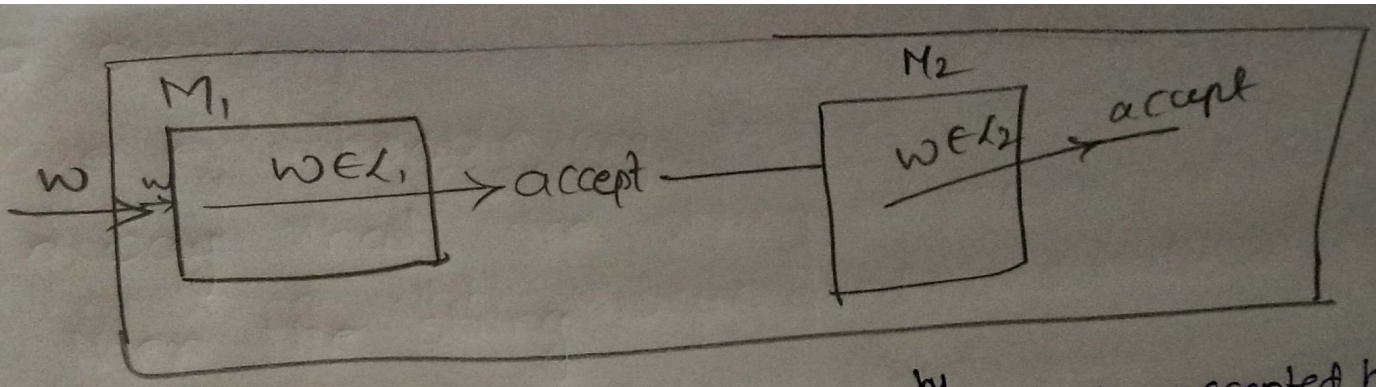
$L_1 \cup L_2 \Rightarrow$ Recursively Enumerable lang.



The machine will accepts both L_1 & L_2 lang

intersection: L_1 & L_2 are recursively enumerable lang then the intersection of L_1 & L_2 is also recursively enumerable lang.

$L_1 \cap L_2 \Rightarrow$ REL



The ~~Strings~~ strings which are accepted by M_1 only accepted by M_2.