Theory of Automata and Formal Language

Lecture-39

Dharmendra Kumar
(Associate Professor)
Department of Computer Science and Engineering United College of Engineering and Research, Prayagraj March 30, 2021

Variations or types of TM

1) Non-deterministic Turing Machine(TM)
2) Multi-tape Turing Machine(TM)
3) Multi-head Turing Machine(TM)
4) Multi-directional Turing Machine(TM)

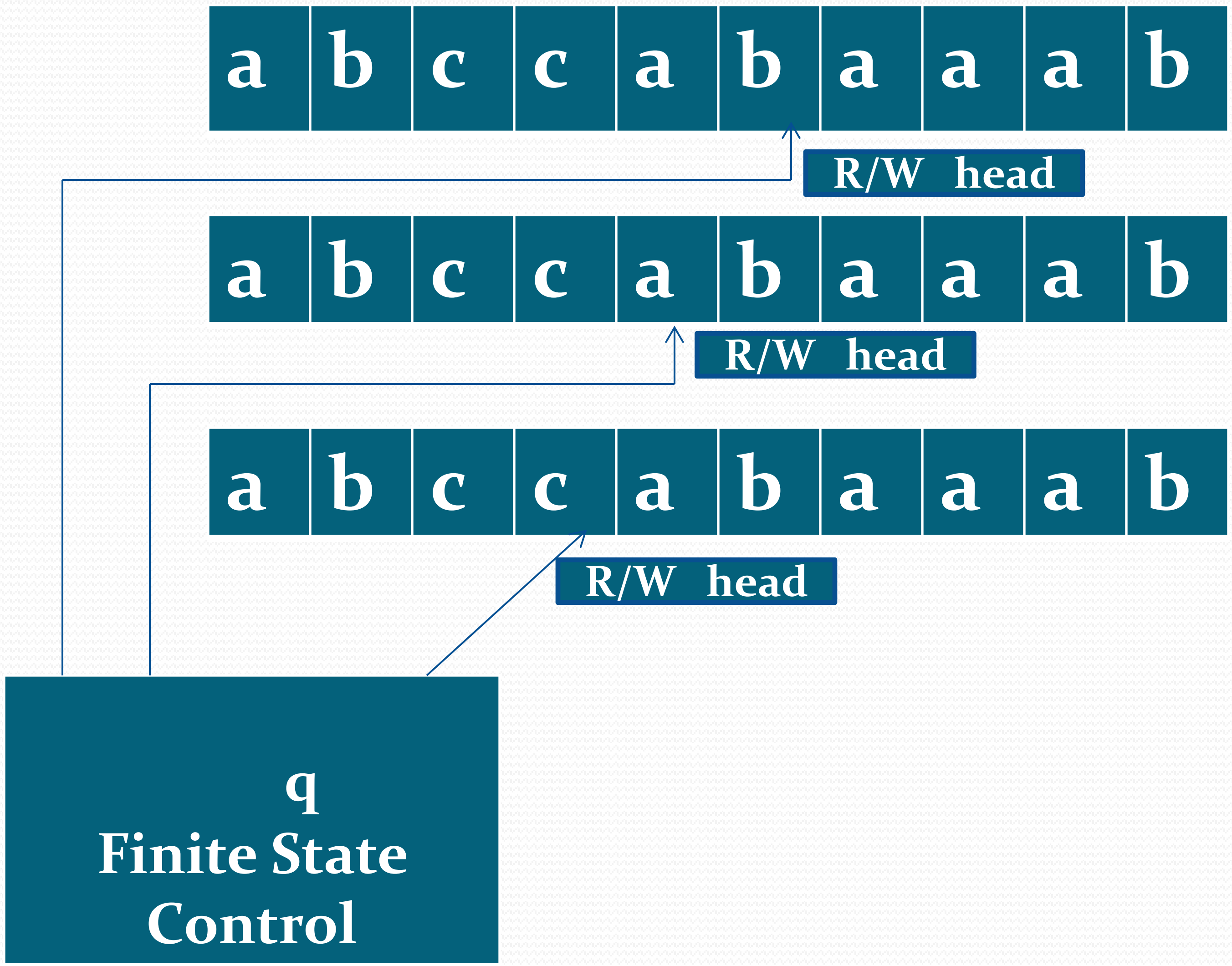
Non-deterministic Turing Machine(TM)

- A non-deterministic TM is a Turing machine which, like nondeterministic finite automata, at any state it is in and for the tape symbol it is reading, can take any action selecting from a set of specified actions rather than taking one definite predetermined action.
- Even in the same situation it may take different actions at different times.
- It differs from deterministic TM only by transition function.
- The transition function of non-deterministic TM is defined as following:-

$$\delta: Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L, R\}}$$

Multi-tape Turing Machine(TM)

Model of TM



Multi-tape Turing Machine(TM)

This type of machine consists of n number of tapes. Since number of tapes is n, therefore the number of heads will also be n.

Transition function will be

$$\delta: Q \times \Gamma^n \rightarrow Q \times \Gamma^n \times \{L, R\}^n$$

Where

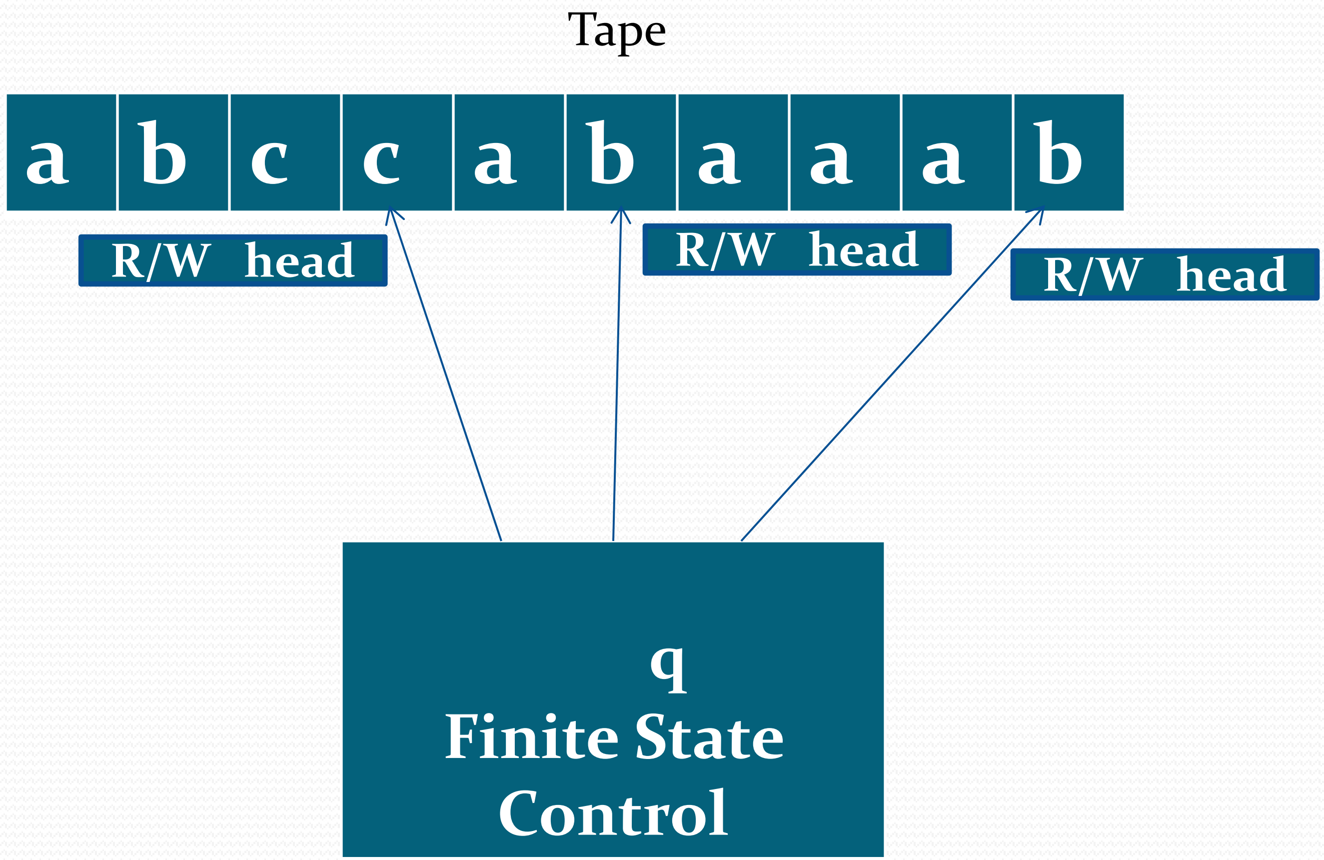
$$\Gamma^n = \Gamma \times \Gamma \times \Gamma \times \dots\dots\dots\dots \times \Gamma \text{(upto n times)}$$

$$\{L, R\}^n = \{L, R\} \times \{L, R\} \times \{L, R\} \times \dots\dots\dots\dots \times \{L, R\}$$

(upto n times)

Multi-head Turing Machine(TM)

Model of TM



Multi-head Turing Machine(TM)

This type of machine consists of one tape with n heads.

Transition function will be

$$\delta: Q\times\Gamma^n \rightarrow Q\times\Gamma^n\times\{L, R\}^n$$

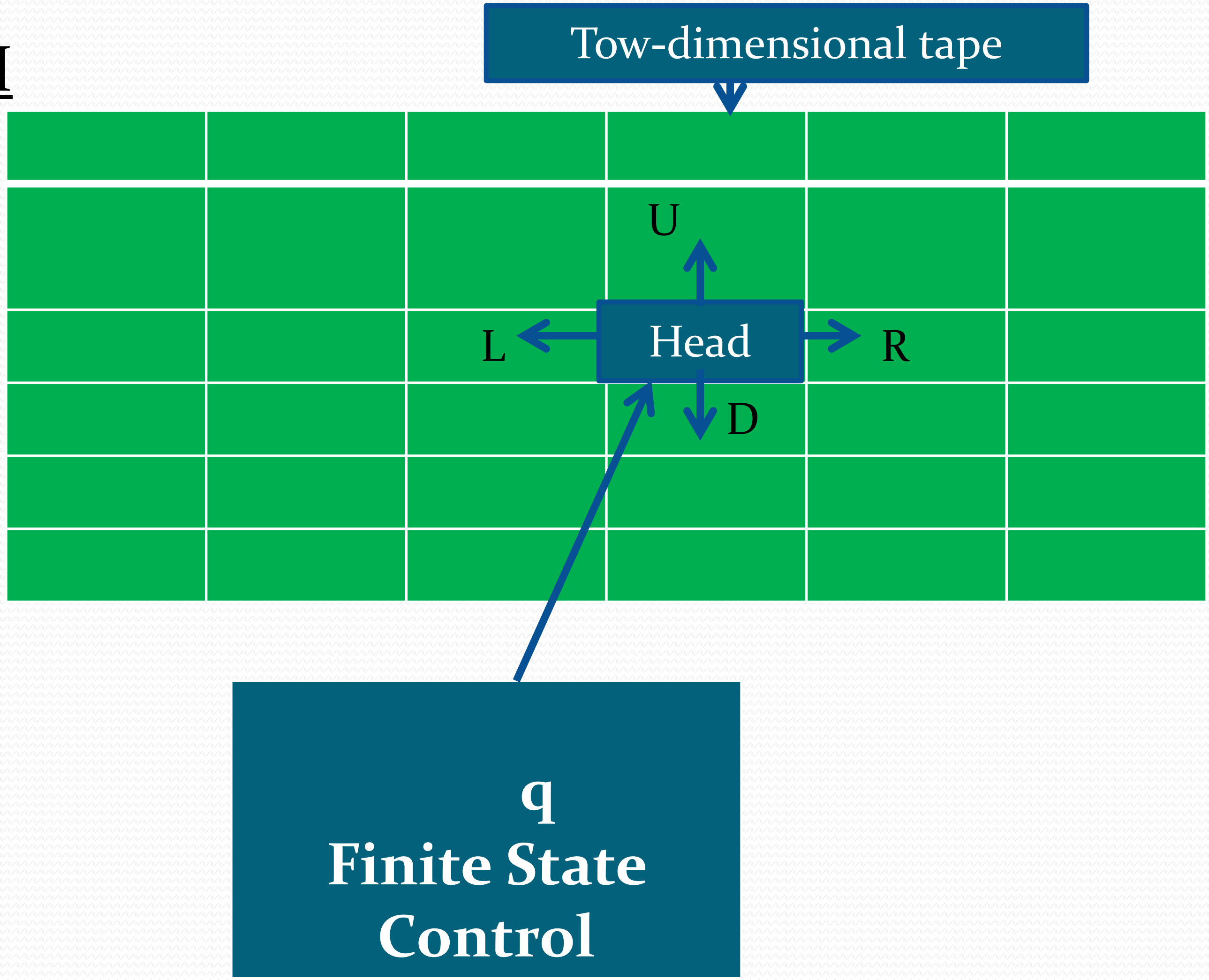
Where

$$\Gamma^n = \Gamma\times \Gamma\times \Gamma\times\ldots\ldots\ldots\ldots \times \Gamma(\text{upto n times})$$

$$\{L, R\}^n = \{L, R\}\times \{L, R\}\times \{L, R\}\times\ldots\ldots\ldots\ldots \times \{L, R\}$$
$$(\text{upto n times})$$

Multi-dimensional Turing Machine(TM)

Model of TM



Multi-dimensional Turing Machine(TM)

- This type of machine consists of one multi-dimensional tape with one heads.
- The head of machine move in many directions.
- If tape is n-dimensional then head move in 2^n directions.
- Transition function of two-dimensional TM is defined as

$$\delta: Q\times\Gamma \rightarrow Q\times\Gamma\times\{L, R, U, D\}$$

Where

L→ Left direction

R→ Right direction

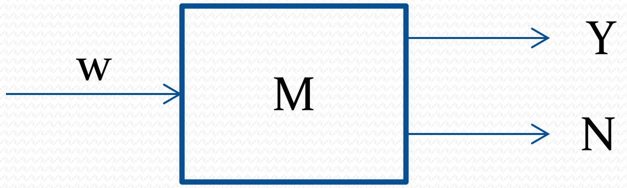
U→ Up direction

D→ Down direction

Recursive and Recursive Enumerable language

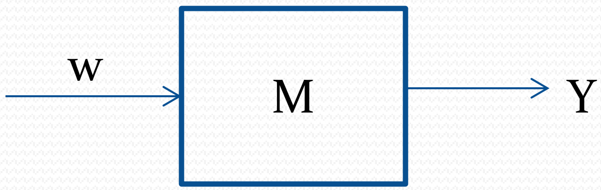
Recursive language:

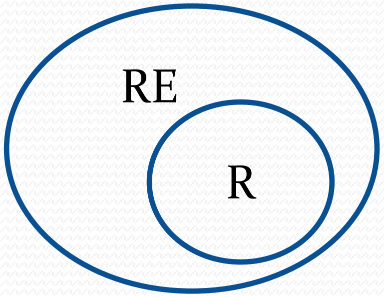
A language L is said to be recursive if there exists a Turing machine M which accepts all the strings w belong into L and rejects all the strings which do not belong into L.



Recursive Enumerable language:

A language L is said to be recursive enumerable if there exists a Turing machine M which accepts all the strings w belong into L and rejects or goes into an infinite loop for all the strings which do not belong into L.



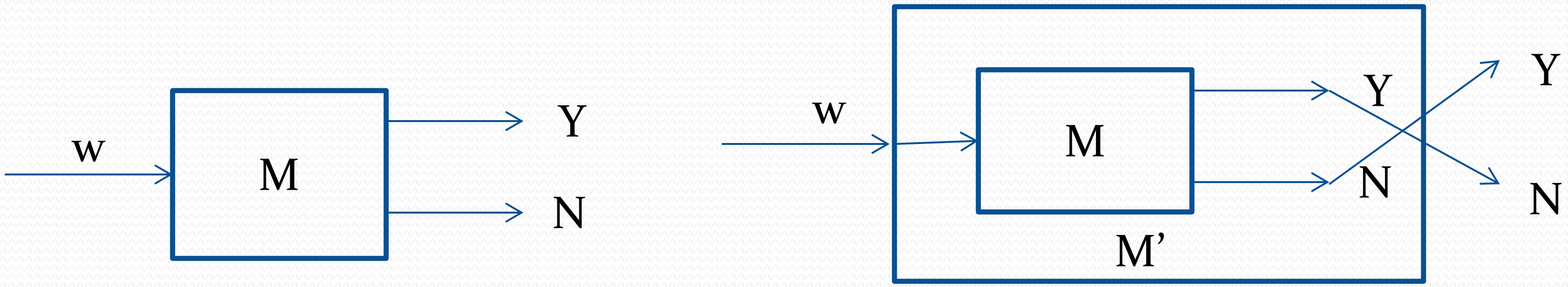


Properties of Recursive and Recursive Enumerable languages

1) If L is recursive language then $\bar{L}$ is also recursive language.
2) If L and $\bar{L}$ are recursive enumerable languages then L will be recursive language.
3) The union of two recursive languages is also recursive i.e. if L_1 and L_2 are recursive then $L_1 \cup L_2$ will be also recursive.
4) The union of two recursive enumerable languages is also recursive enumerable i.e. if L_1 and L_2 are recursive enumerable then $L_1 \cup L_2$ will be also recursive enumerable.
5) The intersection of two recursive languages is also recursive i.e. if L_1 and L_2 are recursive then $L_1 \cap L_2$ will be also recursive.
6) The intersection of two recursive enumerable languages is also recursive enumerablei.e. if L_1 and L_2 are recursive enumerablethen $L_1 \cap L_2$ will be also recursive enumerable .

Theorem:If L is recursive language then complement of L i.e. $\bar{L}$ is also recursive language.

Proof: Since L is recursive language, therefore there exists a TM which accepts all strings belong into L and rejects all strings which do not belong into L. Let this TM is M. Therefore M will be



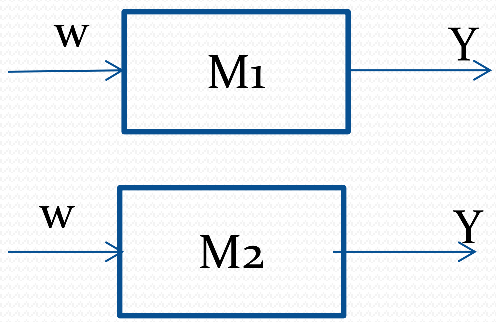
Now, we construct a TM M’ using M as above.

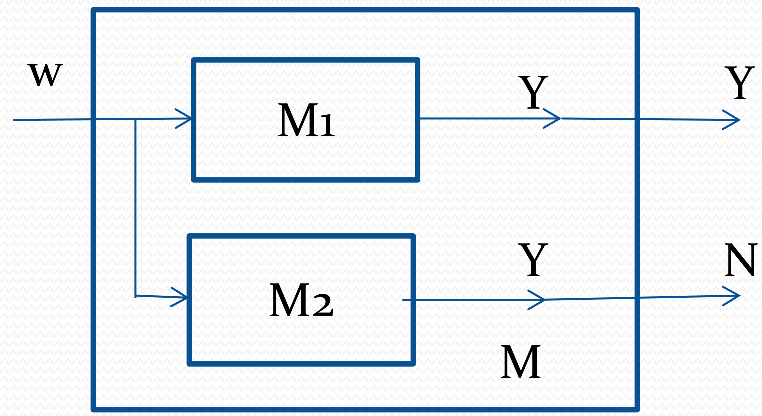
Clearly, if w is accepted by M then w is rejected by M’ and if w is rejected by M then w is accepted by M’. Since L is accepted by M, therefore complement of L i.e. $\bar{L}$ is accepted by M’.

Since there exists a TM M’ corresponding to $\bar{L}$, therefore $\bar{L}$ is recursive language.

Theorem: If L and $\overline{L}$ are recursive enumerable languages then L will be recursive language.

Proof: Since L and $\overline{L}$ are recursive enumerable language, therefore there exists TM M1 and M2 corresponding to L and $\overline{L}$ respectively. M1 accepts all strings belong into L andM2 accepts all strings belong into and $\overline{L}$. These are





Now, we construct a TM M using M1 and M2 as above.

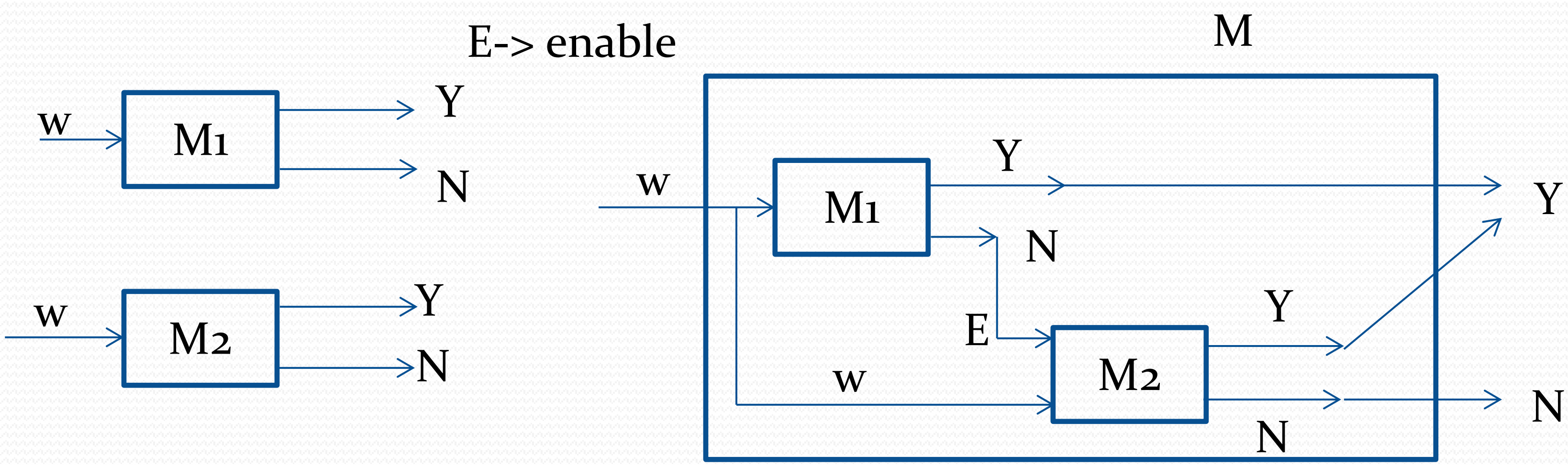
Clearly, if w is accepted by M1 then w is also accepted by M and if w is accepted by M2 then w is rejected by M. That is, if $w \in L$ then it is accepted by M and if $w \notin L$ then it is rejected by M.

Therefore, M is a TM which accepts all strings of L and rejects all strings which are not belong into L.

Hence L is recursive language.

Theorem: The union of two recursive languages is also recursive i.e. if L_1 and L_2 are recursive then $L_1 \cup L_2$ will be also recursive.

Proof: Since L_1 and L_2 are recursive languages then there exists TM M_1 and M_2 corresponding to L_1 and L_2 respectively are of the form:



Consider a string $w \in L_1 \cup L_2$. Then $w \in L_1$ or $w \in L_2$.

If $w \in L_1$ then it is accepted by M_1. therefore it is also accepted by M. If $w \in L_2$ then it is accepted by M_2. therefore it is also accepted by M.

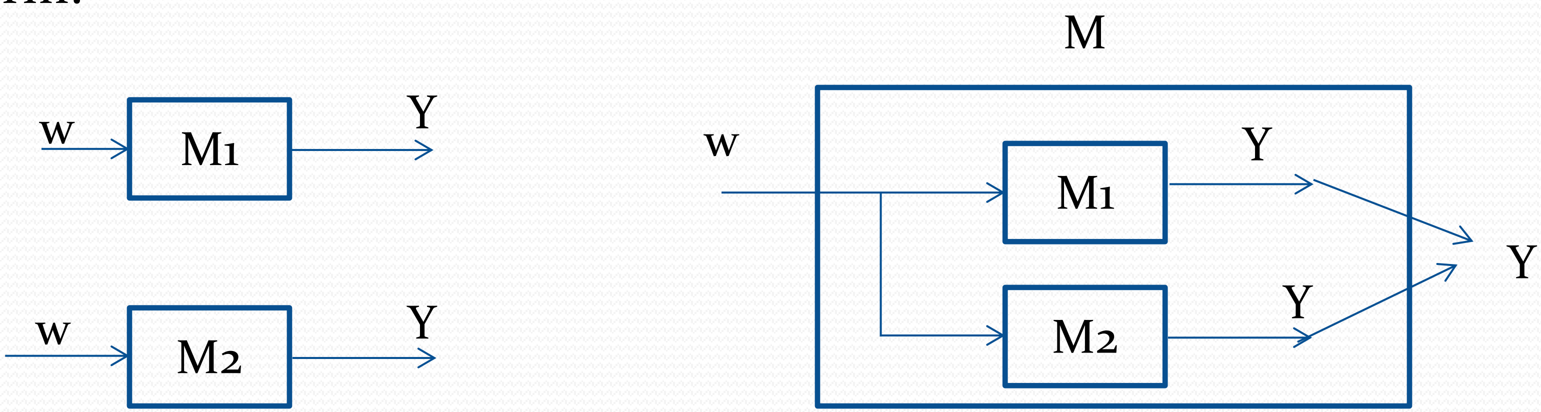
But if $w \notin L_1 \cup L_2$ then w is neither accepted by M_1 nor M_2. therefore it is also not accepted by M.

Hence M is a machine which accepts all strings belong into $L_1 \cup L_2$ and rejects all strings which do not belong into $L_1 \cup L_2$.

Therefore $L_1 \cup L_2$ is recursive language.

Theorem: The union of two recursive enumerable languages is also recursive enumerable i.e. if L1 and L2 are recursive enumerable then L1⋃ L2 will be also recursive enumerable.

Proof: Since L1 and L2 are recursive enumerable languages then there exists TM M1 and M2 corresponding to L1 and L2 respectively are of the form:



Consider a string w ∈ L1⋃ L2. Then w ∈ L_1 or w ∈ L_2.

If w ∈ L_1 then it is accepted by M_1. therefore it is also accepted by M. If w ∈ L_2 then it is accepted by M_2. therefore it is also accepted by M.

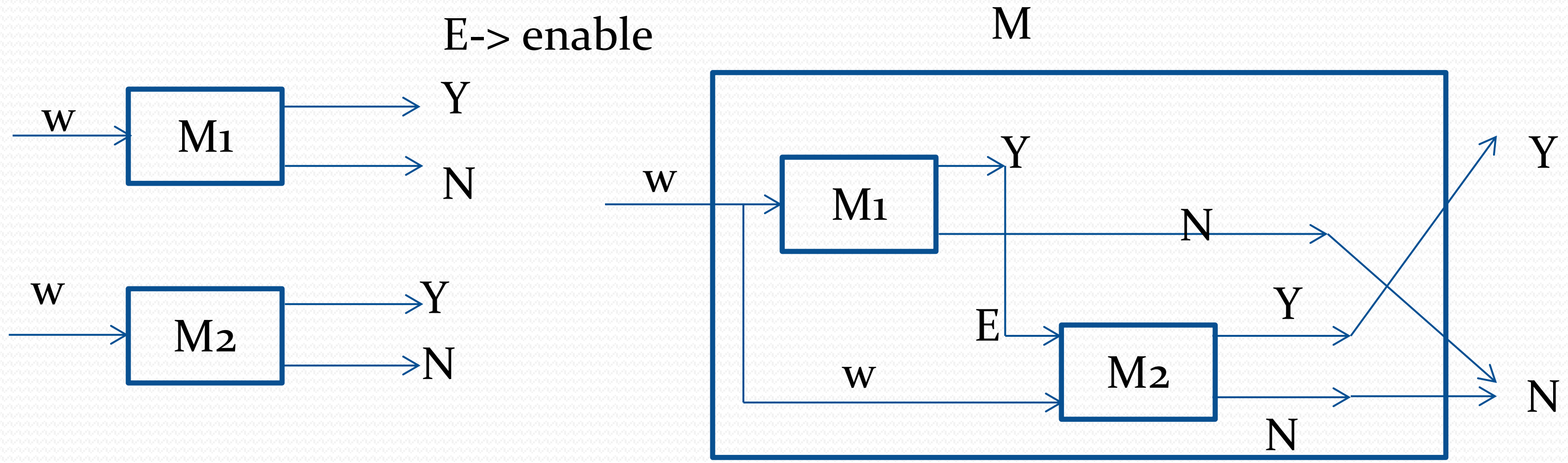
But if w ∉ L1⋃ L2 then w is neither accepted by M_1 nor M_2. therefore it is also not accepted by M.

Hence M is a machine which accepts all strings belong into L1⋃ L2 and rejects all strings which do not belong into L1⋃ L2.

Therefore L1⋃ L2 is recursive language.

Theorem: The intersection of two recursive languages is also recursive i.e. if L1 and L2 are recursive then L1$\cap L_2$ will be also recursive.

Proof: Since L1 and L2 are recursive languages then there exists TM M1 and M2 corresponding to L1 and L2 respectively are of the form:



Consider a string w$\in$ L1$\cap L_2$. Then w $\in L_1$and w $\in L_2$.

Since w $\in L_1$therefore it is accepted by M_1. therefore it is also accepted by M. Since w $\in L_2$therefore it is accepted by M_2. Clearly, therefore it is also accepted by M.

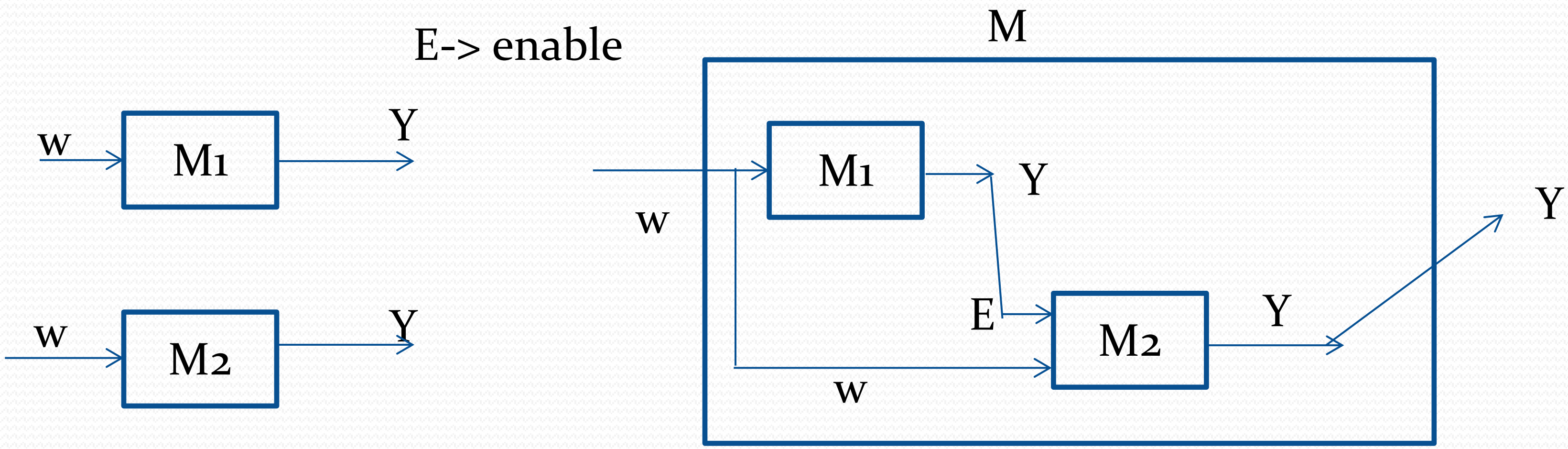
But if w $\notin$ L1$\cap L_2$ then w is either not belong into L1 or not belong into L_2. therefore it is either not accepted by M_1 or not accepted by M_2. Clearly, therefore w is not accepted by M.

Hence M is a machine which accepts all strings belong into L1$\cap L_2$ and rejects all strings which do not belong into L1$\cap L_2$.

Therefore L1$\cap L_2$ is recursive language.

Theorem: The intersection of two recursive enumerable languages is also recursive enumerablei.e. if L1 and L2 are recursive enumerablethen L1$\cap$ L_2 will be also recursive enumerable.

Proof: Since L1 and L2 are recursive enumerable languages then there exists TM M1 and M2 corresponding to L1 and L2 respectively are of the form:



Consider a string w $\in$ L1$\cap$ L_2. Then w $\in$ L_1 and w $\in$ L_2.

Since w $\in$ L_1 and w $\in$ L_2, therefore it is accepted by both M_1 and M_2. Clearly, therefore it is also accepted by M.

But if w $\notin$ L1$\cap$ L_2 then w is either not belong into L_1 or not belong into L_2. In this case, we can not say that w is accepted or not accepted by M_1 or M_2. Clearly, therefore we can also say that w is accepted or not by M.

Hence M is a machine which accepts all strings belong into L1$\cap$ L_2 and rejects or goes into infinite loop for all strings which do not belong into L1$\cap$ L_2.

Therefore L1$\cap$ L_2 is recursive enumerable language.