UNIT-4

Turing Machines:

1. Turing Machine Model:

![Diagram showing a Turing Machine model with a head and a tape divided into cells and of infinite length.]

- A Turing machine $M$ is a 7-tuple, namely $(Q, \Sigma, \Gamma, \delta, q_0, b, F)$ where:
  - $Q$ → finite non-empty set of states
  - $\Sigma$ → finite non-empty set of tape symbols
  - $\Gamma$ → finite non-empty set of input symbols
  - $\delta$ → transition function $[\{(q, v)\} \rightarrow (q', v', D)]$
  - $q_0$ → initial state
  - $F$ → set of final states
  - $b$ → blank

2. Representation of Turing machines:

   (i) Instantaneous description (ID):

   An ID of a Turing machine $M$ is a string $a\beta\gamma$, where $\beta$ is the present state of $M$, the entire input string is split as $\alpha \gamma$, the first symbol of $\gamma$ is the current symbol $\alpha'$ under the R/W head and $\gamma$ has all the subsequent symbols of the input string, and the string $\alpha$ is the substring of the input string formed by all the symbols to the left of $\alpha'$.

   ![Diagram showing left and right sequences with a present state and symbol under R/W head.]

   Repetition of ID

   (ii) Transition diagrams (contains states and edges in the form of $(q, y, D)$)

   (iii) Transition table (contains state and tape symbols)
3. **Language Acceptability by Turing Machine**:

   - A language is said to be accepted by a Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, F, b)$ if $\delta(q_0, \varepsilon, \varepsilon) = q_1, \delta(q_1, \varepsilon, \varepsilon) = q_2$ for some $q_1, q_2 \in F$ and $q_0, q_1, q_2 \in Q$.

   - $M$ does not accept $w$ if the machine halts in a non-accepting state or does not halt.

4. **Design of Turing Machines**:

   i. Basic guidelines for designing a Turing machine:

   - The fundamental objective in scanning by the R/W head is to 'know' what to do in the future. The machine must remember the past symbols scanned. The Turing machine can remember this by going to a new, unique state.

   - The number of states must be minimized. This can be achieved by changing the states only when there is a change in the written symbol or when there is a change in the movement of the R/W head.

5. **Universal Turing Machine and Other Modifications**:

   - A universal Turing machine (UTM) is a Turing machine $T_u$ that works as follows. It is assumed that to receive an input string of the form $e(T)\ v(z)$, where $T$ is an arbitrary TM, $v$ is a string over the input alphabet of $T$, and $e$ is the encoding function whose values are strings in $\{0, 1\}^*$. The computation performed by $T_u$ on this input string satisfies these two properties:

   1. $T_u$ accepts the string $e(T)\ v(z)$ if and only if $T$ accepts $z$.
   2. If $T$ accepts $z$ and produces output $y$, then $T_u$ produces output $y$.

   **Modifications**:

   1. Increasing number of R/W heads.
   2. Making the tape two or three-dimensional.
   3. Adding a special purpose memory, such as 'stacks' or special purpose registers.
Church's Hypothesis:
Hypothesis means proving certain facts. The church's hypothesis on church's turing thm. can be stated as, "The assumption that the intuitive notion of computable functions can be identified with partial recursive functions." However, this hypothesis can not be proved. The computability of recursive functions is based on following examples:
1. Each elementary function is computable.
2. Let \( f \) be the computable function and \( g \) be the other function which can be obtained by applying an elementary operation to \( f \), then \( g \) becomes a computable function.
3. Any function becomes computable if it is obtained by (1) & (2).

Composite and Iterated TM:
When more than one TM combined together to solve the given problem, such a TM is called composite TM.
The iterated TM is a special case of composite TM in which the output of a TM is given as input to itself.

Turing Machine as Enumerators:
Enumerators are a Turing Machines with an output printer.
An enumerator \( E \) is a 6-tuple \((Q, \Sigma, \Gamma, \delta, q_0, q_{\text{print}})\) where
- \( Q \) → finite set of states
- \( \Sigma \) → print tape alphabet
- \( \Gamma \) → work tape alphabet
- \( \delta \) → transition function \((Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, \alpha\} \times \{L, R\})\)
- \( q_0 \) → Initial state
- \( q_{\text{print}} \) → final state

TWO TAPE DEVICE
\[ \text{write only on tape} \]
\[ \text{work tape} \]
9. \textit{Recursively Enumerable language}:
A language \( L \subseteq \Sigma^* \) is recursively enumerable if there exists a TM \( M \), such that \( L = T(M) \).

10. \textit{Recursive language}:
A language \( L \subseteq \Sigma^* \) is recursive if there exists some TM \( M \) that satisfies the following two conditions:
(i) If \( w \in L \) then \( M \) accepts \( w \) (that is, reaching an accepting state on processing \( w \)) and halts.
(ii) If \( w \notin L \) then \( M \) eventually halts, without reaching an accepting state.

\begin{center}
\textbf{Recursive language} \subset \textbf{Recursive enumerable language}
\end{center}

11. \textit{Properties of recursive & recursively enumerable language}:
(1) If \( L \) is a recursive language, then \( L \) is also a recursive language.
(2) If both \( L \) and \( I \) are recursively enumerable language, then \( L \) and \( I \) is also recursive.