Formal Language and Automata Theory

Chapter six

Turing Machine

The Standard Turing Machine

- Turing machine's storage is actually quite simple. It can be visualized as a single, one-dimensional array of cells, each of which can hold a single symbol.
- This array extends indefinitely in both directions and is therefore capable of holding an unlimited amount of information. The information can be read and changed in any order.
- we will call such a storage device a tape because it is analogous to the magnetic tapes used in actual computers.

Definition

- A Turing machine is an automaton whose temporary storage is a tape.
- A Turing Machine (TM) is a mathematical model which consists of an infinite length tape divided into cells on which input is given. It consists of a head which reads the input tape.

Con’t

- A state register stores the state of the Turing machine. After reading an input symbol, it is replaced with another symbol, its internal state is changed, and it moves from one cell to the right or left.
- If the TM reaches the final state, the input string is accepted, otherwise rejected.

The key features of the Turing machine model

1. A finite amount of internal state.
2. An infinite amount of external data storage.
3. A program specified by a finite number of instructions in a predefined language.
4. Self-reference: the programming language is expressive enough to write an interpreter for its own programs.

Formal definition

- A TM can be formally described as a 7-tuple (Q, $\mathbf{\Gamma}$, Σ, δ, q_0, B, F) where:
  - **Q** is a finite set of states
  - **Γ** is the tape alphabet
  - **Σ** is the input alphabet
  - **δ** is a transition function; $\delta : Q \times \mathbf{\Gamma} \rightarrow Q \times \mathbf{\Gamma} \times \{\text{Left\_shift}, \text{Right\_shift}\}$.
  - $\mathbf{q_0}$ is the initial state
  - **B** is the blank symbol
  - **F** is the set of final states

Comparison with the previous automaton:

- If we compare finite automata with pushdown automata, we see that the nature of the temporary storage creates the difference between them.
- If there is no storage, we have a finite automaton; if the storage is a stack, we have the more powerful pushdown automaton.
- Extrapolating from this observation, we can expect to discover even more powerful language families if we give the automaton more flexible storage.

Accepted Language and Decided Language

- A TM accepts a language if it enters into a final state for any input string **w**. A language is recursively enumerable (generated by Type-0 grammar) if it is accepted by a Turing machine.
- A TM decides a language if it accepts it and enters into a rejecting state for any input not in the language. A language is recursive if it is decided by a Turing machine.
- There may be some cases where a TM does not stop. Such TM accepts the language, but it does not decide it.

Universal Turing Machine

- The key property of Turing machines, and all other equivalent models of computation, is universality: there is a single Turing machine U that is capable of simulating any other Turing machine | even those with vastly more states than U.
- In other words, one can think of U as a Turing machine interpreter," written in the language of Turing machines.
- This capability for self-reference (the language of Turing machines is expressive enough to write an interpreter for itself) is the source of the surprising versatility of Turing machines and other models of computation.

The Turing-Church thesis

- *"The languages that can be recognized by an effective procedure are those that are decided by a Turing machine."*
- Justification
  - If a language is decided by a Turing machine, it is computable:
  - If a language is computable, it is decided by a Turing machine: