NOTE

Make sure to check the Automata Theory cheatsheet, too.

1. Intro

A turing machine is a finite automation with unbounded random access memory. The finite state automaton (FSA) provides instructions on an infinite tape, where the input is given and can also be the working space. Every cell contains one character, but some cells are empty. A tape head reads and writes according to the instructions given by the FSA.

2. Formal Definition

A turing machine ™ consists of:

$$(Q, \Sigma, \Gamma, \delta, q_1, q_{Acc}, q_{Rej})$$

where

- ullet Γ is the tape alphabet that included the blank symbol.
- $\Sigma \subseteq \Gamma$ is the input alphabet.
- ullet $\delta \colon Q imes \Gamma o (Q imes \Gamma imes \{L,R\})$ is the transition function.
- $ullet q_1 \in Q$ is the start state.
- ullet $q_{
 m Acc}$ is the accepting state.
- q_{Rej} is the rejecting state.

The **transition function** takes one state and one letter from Γ and returns a **state** of the automaton, a **letter to be written** on the current cell of the tape and the **direction** instructing the tape head where to go, L for left, R for right.

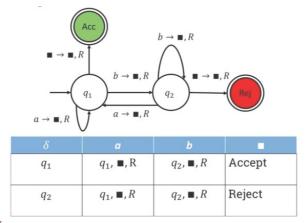


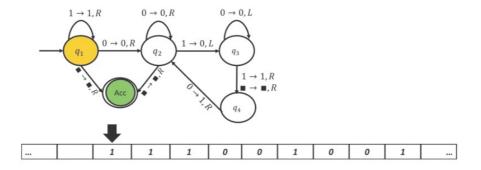
Figure 1. Note that the black box means "blank".

NOTE

Seeing an interactive example makes the behavior of the direction $\{L,R\}$ clearer. See Week 13, "7.201 Turing machines: examples" and "7.202 Designing Turing Machines" in the FCS course.

2.1. Example

Given $w \in (1 \cup 0)^*$, make $w \in 1^*0^*$. For example, given 111001001, make 111110000.



NOTE

For the interactive process, see Week 14, "7.301 The power of Turing machines".

The difference between a Deterministic Finite Automata (DFA) and a Turing Machine (TM) is that a TM may not terminate when the input is completely processed and may process the input several times. A TM always terminate at the accepting or rejecting state, while in DFA the process can pass through those states and continue. Additionally, a TM may manipulate the input, may enter an infinite loop and is deterministic.

4. The Language of Turing Machines

The language of a TM is:

$$\mathscr{L}(M) = \{w \in \Sigma^* \mid M \;\; ext{accepts} \;\; w\}$$

If $w \in \mathcal{L}(M)$, M reached accept state. If $w \notin \mathcal{L}(M)$, M does not reach accept state (either it reaches reject state or enters an infinite loop). A language is **recognizable** if it is accepted by a TM, where the TM is called the **recognizer** of $\mathcal{L}(M)$.

A TM that does not enter an infinite loop is called a **decider**. The language is decidable if it is **accepted** by the decider.

4.1. Halting Problem

RE ("recursively enumerable") is a class of all recognizable languages R is a class of all decidable languages.

Additionally:

$$R \subseteq RE$$

In other words, every decider is a recognizer, but not the other way around. The **halting problem** states that we cannot determine whether an arbitrary TM and an input will eventually halt or run forever.

4.2. Language Hierarchy

$$RL \subseteq CFL \subseteq R \subseteq RE \subseteq all languages$$

where "RL" refers to Regular Languages and "CFL" refers to Context-Free Languages.

4.2.1. Chomsky Hierarchy

Grammar	Languages	Automaton	Example
Type-0	Recursively enumerable	Turing machine	
Type-1	Context-sensitive	Turing machines with bounded tape	$a^nb^nc^n$
Type-2	Context-free	Push-down	a^nb^n
Туре-3	Regular	Finite state	a^*b^*

Last updated 2023-01-21 18:35:01 UTC