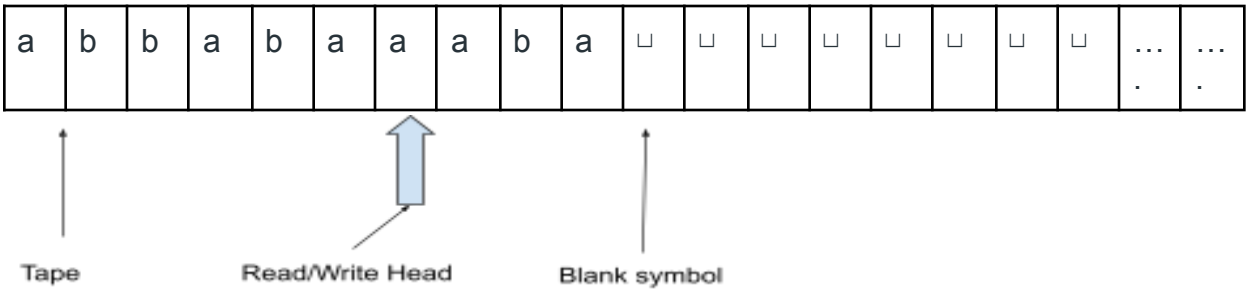
Cairo University
Faculty of Computers & Artificial Intelligence
Theory of Computations

Lab#10

Turing Machine

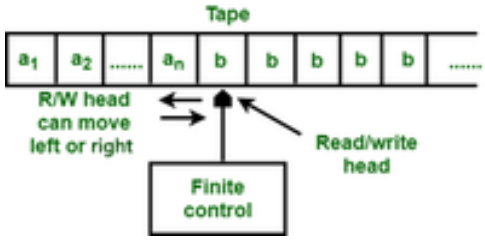
A Turing machine is a finite automaton that can read, and write symbols on an infinitely long tape. It is used to study the properties of algorithms and to determine what problems can and cannot be solved by computers and provide a way to model the behavior of algorithms.



Tape alphabets: $\Sigma = \{a, b\}$

The blank ␣ is a special symbol. $␣ \notin \Sigma$

The blank is used to fill the infinite tape.



The finite control is the program, it is deterministic and it is the portion that controls the turing machine.

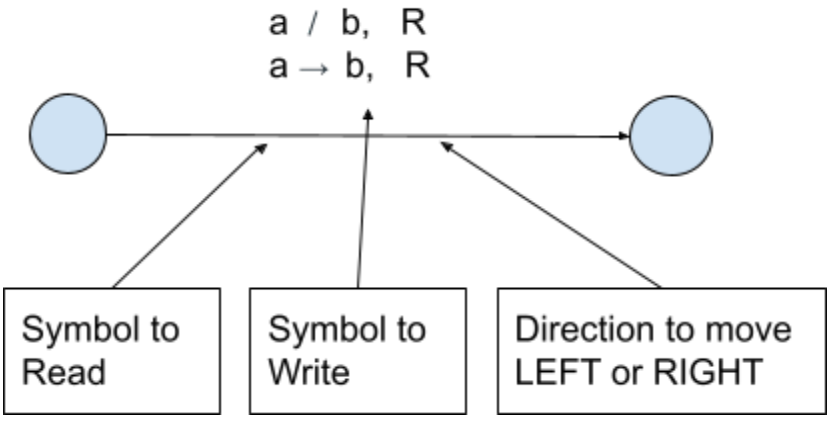
Operations on the Tape:

1) Read a symbol below the tape head.

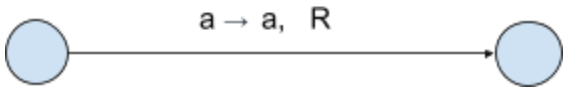
2) Update/ Write a symbol below the tape head.

3) Move the tape head one step LEFT.

4) Move the tape head one step RIGHT.



If you don't want to update the cell, just write the same symbol.



TM has 1 initial state and 2 final states (accept state and reject state).

A TM is expressed as a 7-tuple **(Q, ∑, Γ, δ, q0, b, F)** where:

- **Q** → a finite set of states.
- **∑** → the input alphabet (symbols which are part of input alphabet).
- **Γ** → the tape alphabet (symbols which can be written on Tape).
- **δ** → a transition function which maps Q × ∑ → Γ × (L/R) × Q.
- **q0** → the initial state
- **b** → blank symbol
- **F** → the set of final states (accept state and reject state).

Ex: **δ(q0, a) → (q1, y, R)**

When we are in state q0 and have an input symbol a, we will go to state q1 and write y on the tape and move 1 step to the Right.

Example 1:

Design a turing machine that accepts even palindromes over the alphabet:

$$\Sigma = \{a, b\}$$

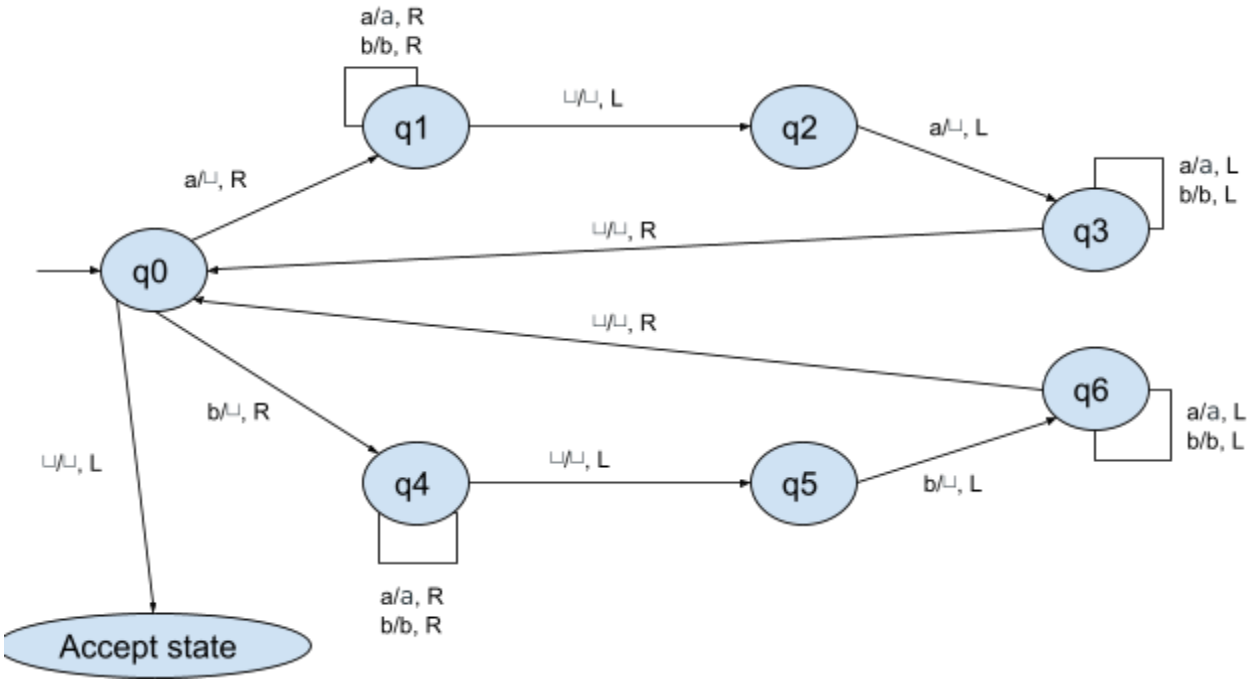
eg:

a	b	a	a	b	a	␣	␣	……

Solution:

Algorithm

- Read first symbol and replace it with ␣
- Move Right until find last symbol (before a ␣)
- If they are they same, replace it with a ␣
- Then move Left until you reach the first symbol (after a ␣).
- Repeat these steps until there is no more symbols (all ␣).



Example 2:

Design a turing machine for the language L = { $0^n 1^n 2^n$ | n≥1 }

Σ = {0, 1, 2}

eg:

0	0	1	1	2	2	⊔	⊔	……

Solution:

Algorithm

- First replace a 0 from front by X.
- Keep moving Right till you find a 1 and replace it by Y.
- Keep moving Right again till you find a 2, replace it by Z.
- Move Left till you find X.
- Repeat these steps again.