Theory of Computing

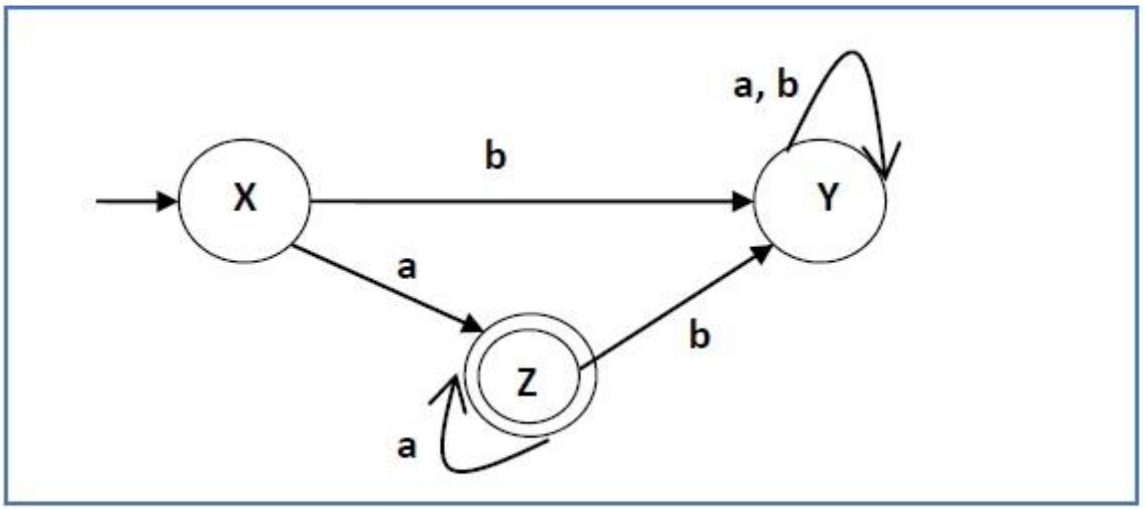
CSE – 203

DFA complement, Turing Machine

DFA Complement

If (Q, ∑, δ, q_0, F) be a DFA that accepts a language L, then the complement of the DFA can be obtained by swapping its accepting states with its non-accepting states and vice versa.

We will take an example and elaborate this below −

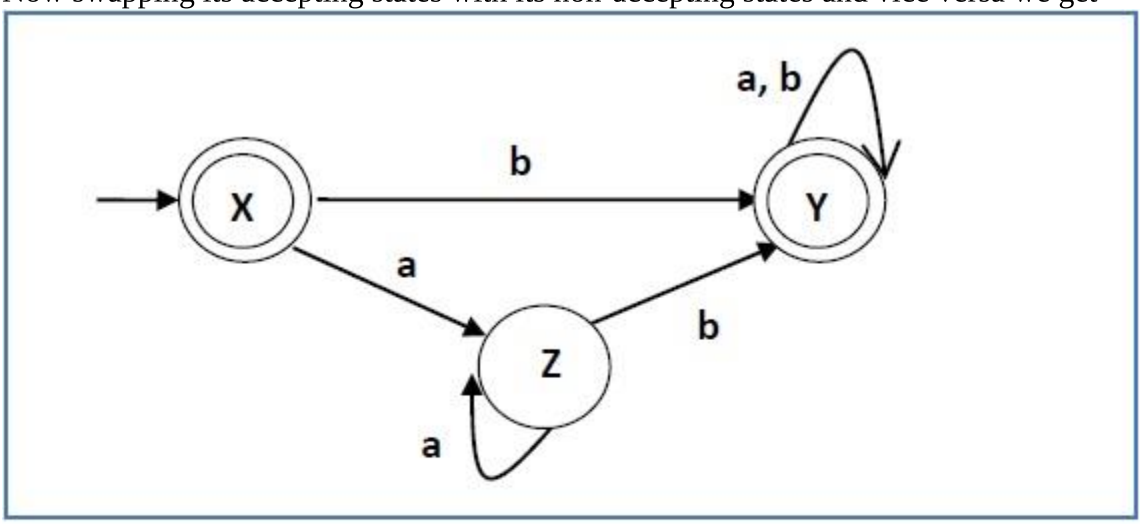


This DFA accepts the language

L = {a, aa, aaa , ... } over the alphabet ∑ = {a, b}

So, RE = a^+.

Now swapping its accepting states with its non-accepting states and vice versa we get



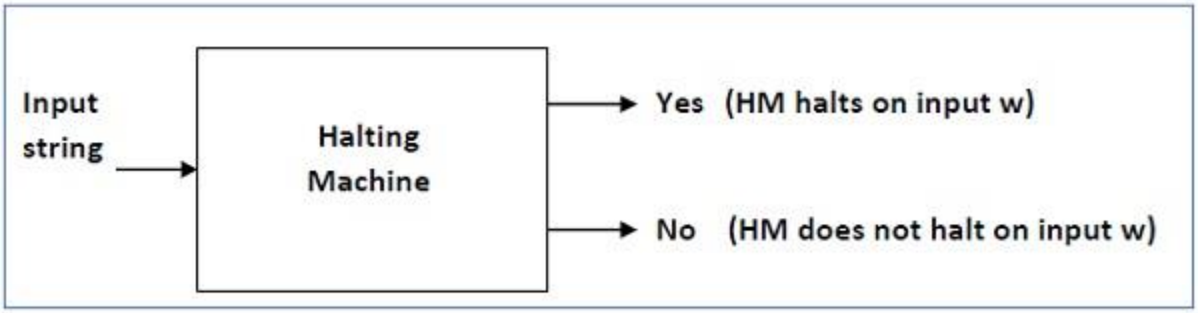
This DFA accepts the language

L' = {ε, b, ab ,bb,ba, ... } over the alphabet ∑ = {a, b}

Turing Machine Halting Problem
Input − A Turing machine and an input string **w**.

Problem − Does the Turing machine finish computing of the string **w** in a finite number of steps? The answer must be either yes or no.

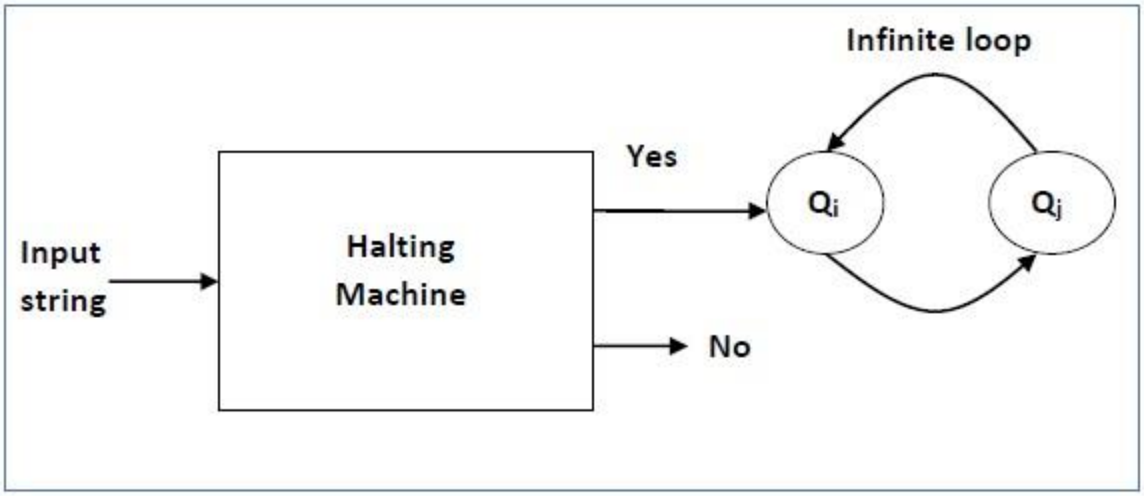
Proof − At first, we will assume that such a Turing machine exists to solve this problem and then we will show it is contradicting itself. We will call this Turing machine as a *Halting machine* that produces a 'yes' or 'no' in a finite amount of time. If the halting machine finishes in a finite amount of time, the output comes as 'yes', otherwise as 'no'. The following is the block diagram of a Halting machine −



Now we will design an **inverted halting machine (HM)'** as −

- If **H** returns YES, then loop forever.
- If **H** returns NO, then halt.

The following is the block diagram of an 'Inverted halting machine' –



Further, a machine **$(HM)_2$** which input itself is constructed as follows −

- If $(HM)_2$ halts on input, loop forever.
- Else, halt.

Here, we have got a contradiction. Hence, the halting problem is **undecidable**.