8.2 Transitional Representation of Turing Machine

The mathematical notation for a TM is $(Q, \Sigma, \Gamma, \sigma, q0, B, F)$. In a TM, the transitional function σ consists of the form $Q \times \Gamma \rightarrow (Q \times \Gamma \times \{L, R, H\})$, where the first is a present state and the second is the present input (single character $\in \Gamma$). The transition produces a state, a symbol $\in \Gamma$ written on the input tape, and the head moves either left or right or halts.

In the graphical notation of the TM, there are states. Among them, a circle with an arrow indicates a beginning state and a state with double circle indicates a final state, where the machine halts finally. The state transitions are denoted by arrows. The labels of the state transitions consist of the input symbol, the symbol written on the tape after traversal, and the direction of movement of the read–write head (left, right, or halt). The following are some examples of the TM with the transitional representation.

Example 8.16

Design a TM by the transitional notation for the language $L = a^n b^n$, where $n > 0$.

Solution: When the machine traverses 'a', replace that 'a' by X and traverse right to find the leftmost 'b'. Replace that 'b' by 'Y' and traverse left to find the second 'a'. The second 'a' exists after 'X', which was replaced first for the fi rst 'a'. By this process, when n number of 'a' and n number of 'b' are traversed and replaced by X and Y, respectively, then by getting a blank (B) the machine halts. The transitional representation of the TM is given in Fig. 8.2 .

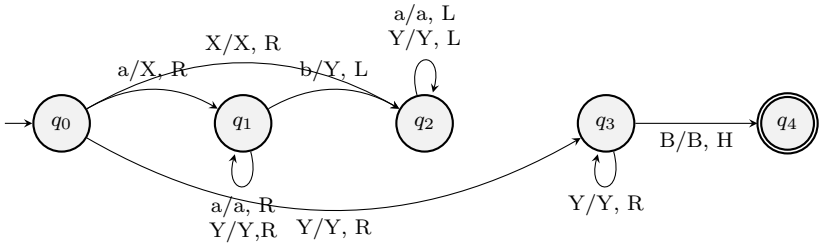


Fig. 8.2 *Transitional Representation of Turing Machine*

Example 8.17

Design a TM by transitional notation for the language L ={set of all palindrome over a,b}.

Solution: A palindrome can be of two types, namely, an odd palindrome and an even palindrome. A null string is also a palindrome. There are three paths to reach to the final state with halt from q1 , the beginning state. This is described in Fig. 8.3.

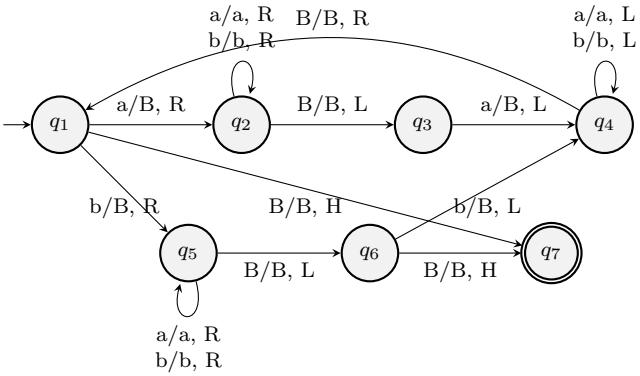


Fig. 8.3 *Turing Machine for Set of All Palindrome Over a, b*

8.3 Non-deterministic Turing Machine

The concept of non-determinism is clear as we have already discussed about NDFA or NDPDA in the earlier chapters. There is also a non-deterministic TM. A non-deterministic TM is defi ned as $(Q, \Sigma, \Gamma, \sigma, q0, B, F)$, where σ is $Q \times \Gamma \rightarrow 2^{QX}\Gamma^{X\{L,R\}}$.

Let there exist a transitional function $\sigma(q0, 0) \rightarrow (q0, 0, R), (q1, 1, R)$.

For traversing the input symbol, we have to consider two transitions and construct the following transitions accordingly. The NTM is more powerful than the DTM. An NTM T_1 can be simulated to an equivalent DTM T_2. One of the methods is to construct a computation tree of the NTM and perform a breadth first search of the computation tree from the root until the halt state is reached. At each level of the constructed tree, T_2 uses the transitional functions of T_1 to each confi guration at that level and computes its children. These children are stored on a tape for the confi guration of the next level. Among the children, if the halting state exists then T_2 halts by accepting the strings. T_2 accepts means that one branch of T_1 accepts it. It can be said that T_2 accepts a string if and only if T_1 accepts it.

Any language accepted by an NTM is also accepted by a DTM. But the time taken by a DTM to accept a language is more than the time taken by an NTM. In most of the cases, it is exponential in length. A famous unsolved problem in computer science, i.e., the P = NP problem, is related to this issue.

Example 8.18

Construct a TM over $\{a, b\}$ which contains a substring abb.

Solution: In regular expression, it can be written as $(a, b)^*abb(a, b)^*$. In this expression, the substring is important. The string may be abb only. In that case, the machine gets 'a' as input in the beginning state. Before traversing the substring 'abb', there is a chance to traverse 'a' of $\{a, b\}^*$. The transitional functions are

$$\sigma(q1, a) \rightarrow (q1, a, R), (q2, a, R)$$
$$\sigma(q1, b) \rightarrow (q1, b, R)$$
$$\sigma(q2, b) \rightarrow (q3, b, R)$$
$$\sigma(q3, b) \rightarrow (q4, b, R)$$
$$\sigma(q4, a) \rightarrow (q4, a, R)$$
$$\sigma(q4, b) \rightarrow (q4, b, R)$$
$$\sigma(q4, B) \rightarrow (qf, B, H)$$

In state q_1 for input 'a', there are two transitional functions. So, it is a non-deterministic TM.

8.4 Conversion of Regular Expression to Turing Machine

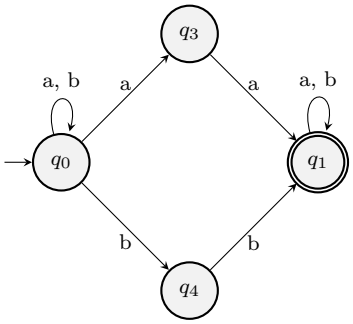
Regular expression can be converted to an equivalent TM. The process is as follows.

1. Convert the regular expression to an equivalent automata without the $\in$ move.
2. Change both the initial and fi nal states of the automata to an intermediate state.
3. Insert a new initial state with a transition (B, B, R) to the automata's initial state.
4. Convert the transitions with label 'ip' to (ip, ip, R).
5. 5. Insert a new fi nal state with a transition (B, B, R) from the automata's fi nal state to the new fi nal state.

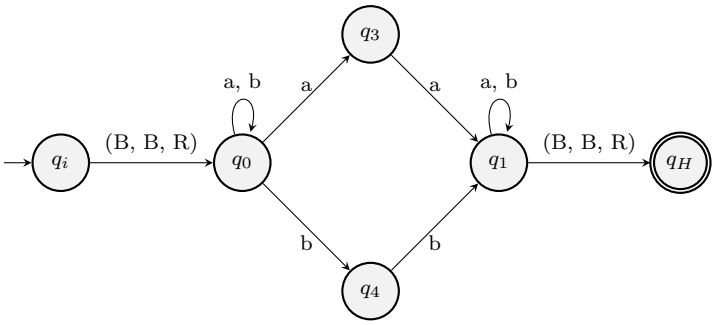
Example 8.19

Construct a TM for the regular expression $(a+b)^*(aa+bb)(a+b^*)$.

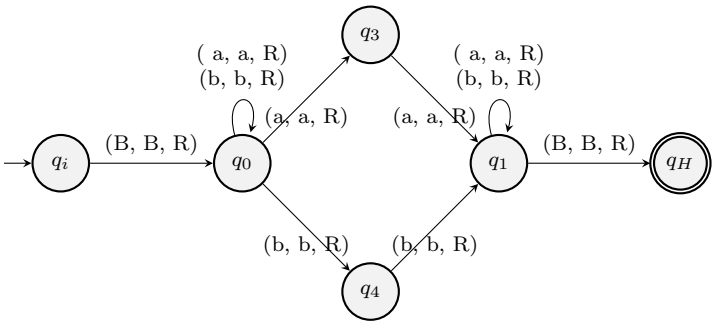
Solution: The finite automata accepting the regular expression is



Inserting a new initial state q_i and final state q_H, the fi nite automata becomes



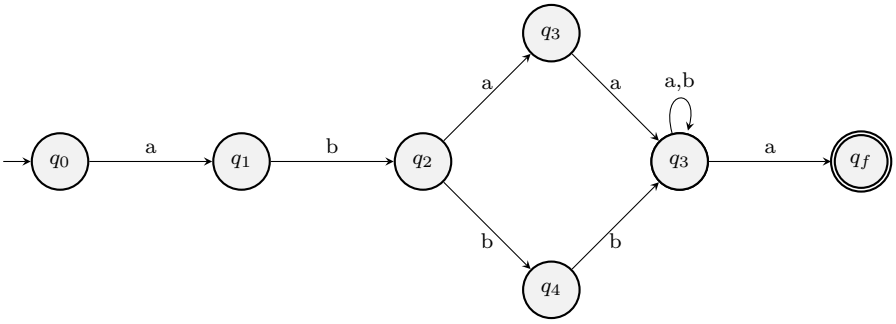
Converting the levels of the inputs, the TM becomes



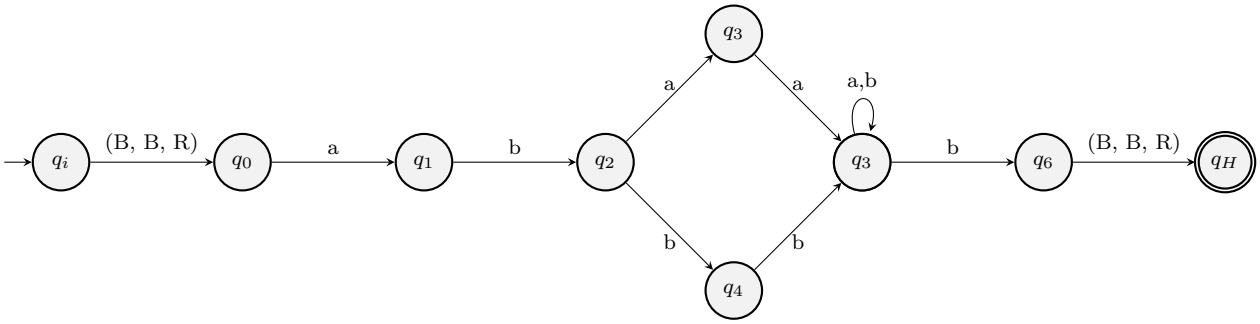
Example 8.20

Construct a TM for the regular expression $ab(aa + bb)(a + b)^*b$.

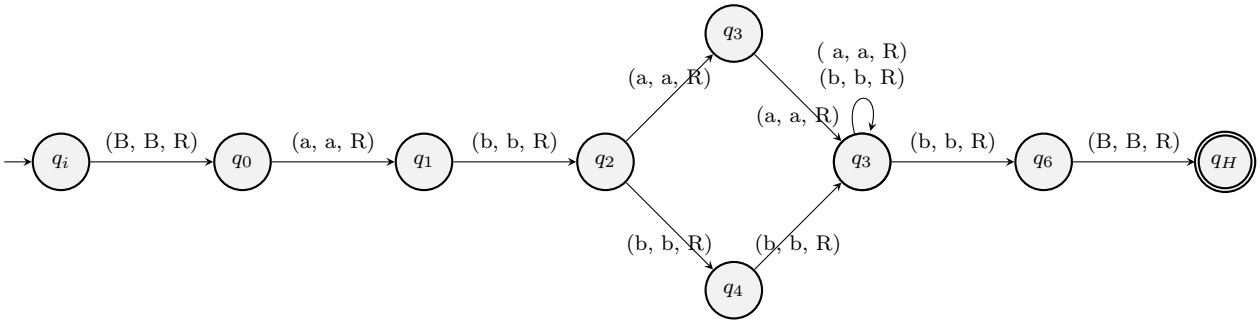
Solution: The finite automata accepting the regular expression is



Inserting a new initial state q_i and final state q_H, the finite automata becomes



Converting the levels of the inputs, the TM becomes



Example 8.21

Design a TM which converts $(0,1)_*011(0,1)^*to(0,1)^*100(0,1)^*$.

Solution: The problem is to design a TM which searches the substring 011 from a string of $0, 1$, and then it replaces 011 by 100. The design is done by the following way.

i) Convert $(0,1)^*011(0,1)^*$ to the accepting DFA.

ii) Convert it to the corresponding TM.

iii) Modify the TM by modifying the transitions of 011 by 100.