Exercise (Example 8.2 from textbook)

Construct a Turing Machine accepting the language 
\[ \{0^n1^n \mid n \geq 1\} \]

Solution

The idea is that the TM \( M \) that we construct needs the leftmost 0, turns it into \( X \), and moves right until it reaches a 1, that is turned into \( Y \). Then the head moves left again to the leftmost 0 (on the right to a \( X \)), and starts again until all 0's and 1's are turned into \( X \)'s and \( Y \)'s respectively.

If the input is not in \( 0^*1^* \), \( M \) will fail to find a move and it won’t accept. If \( M \) changes the last 0 and the last 1 in the same round, it will go into the final state and accept.

\[ Q = \{ q_0, q_1, q_2, q_3, q_4 \} \]
\[ \Sigma = \{ 0, 1 \} \]
\[ \Gamma = \{ 0, 1, X, Y, \# \} \] (\# denotes blank symbol)

\[ q_0 : \text{start state} \]
\[ f = \{ q_f \} \]

In \( q_0 \) is the state in which \( M \) is when the head precedes the leftmost 0. In state \( q_1 \), \( M \) moves right skipping 0's and 1's until it gets to a 1. In state \( q_2 \), \( M \) moves left while skipping \( Y \)'s and \( 0 \)'s again, until it gets to a \( X \) and goes again in \( q_0 \).
Starting from $q_0$, if a $Y$ is read instead of a $0$, H goes in $q_3$ and moves right; if a $1$ is found, then there are more $1$'s then $0$'s; if a $0$ is read, then the initial string is accepted (transition to $q_4$).

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>X</th>
<th>Y</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>$(q_1, X, R)$</td>
<td>—</td>
<td>—</td>
<td>(q₃,Y,R)</td>
<td>—</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$(q_1, 0, R)$</td>
<td>(q₂, Y, L)</td>
<td>—</td>
<td>(q₁, Y, R)</td>
<td>—</td>
</tr>
<tr>
<td>$q_2$</td>
<td>(q₂, 0, L)</td>
<td>—</td>
<td>(q₅, X, R)</td>
<td>(q₂, Y, L)</td>
<td>—</td>
</tr>
<tr>
<td>$q_3$</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>(q₃, Y, R)</td>
<td>(q₄, $\delta$, R)</td>
</tr>
<tr>
<td>$q_4$</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

**Exercise**
Show the computation of the TM above when the input string is:

(a) 00
(b) 000111

**Solution**

(a) $q_0 00 \rightarrow X q_1 0 \rightarrow X 0 q_4$

and the TM halts.

(b) $q_0 000111 \rightarrow X q_1 00111 \rightarrow X 0 q_3 0 111 \rightarrow$

$X 0 X 0 q_1 111 \rightarrow X 0 q_2 0 111 \rightarrow X q_2 0 Y 11 \rightarrow q_2 X 0 X Y 11 \rightarrow$

$q_2 X 0 Y 1 \rightarrow X X q_2 0 Y 1 \rightarrow X 0 q_4 Y 1 \rightarrow X X 0 q_2 Y 1 \rightarrow$

$X X 0 q_4 Y 1 \rightarrow X X q_4 Y 1 \rightarrow X X X q_4 Y 1 \rightarrow X X Y q_5 Y 1 \rightarrow$

$X X q_2 Y Y \rightarrow X X q_2 X Y Y \rightarrow X X X q_6 Y Y \rightarrow X X Y q_3 Y Y \rightarrow$

$X X Y q_3 Y \rightarrow X X X Y Y q_9 \rightarrow X X X Y Y Y q_{10}$
Exercise (8.2.3 from textbook):

Design a Turing Machine that takes as input a number \( N \) in binary and turns it into \( N+1 \) (in binary); the number \( N \) is preceded by the symbol \( \$ \), which may be destroyed during the computation. For example, \( \$111 \) is turned into \( 1000 \); \( \$1001 \) is turned into \( \$1010 \).

Solution

The idea is to toggle the rightmost digit, and, from right to left, all consecutive 1's until we get to the first 0 (which is also toggled). If there is no 0 to be toggled, a 1 is added on the left of the first digit (i.e., in place of the \( \$ \)).

We need three states, where only \( q_2 \) is the final state; we briefly describe what the TM does in the different states.

\[
\begin{array}{c|ccccc}
& \$ & 0 & 1 & 1 & \text{end} \\
\hline
q_0 & (q_0, R, R) & (q_0, 0, R) & (q_0, 1, R) & (q_2, b, L) & \\
q_1 & (q_2, 1, L) & (q_2, 1, L) & (q_2, 0, L) & - & \\
q_2 & - & - & - & - & - \\
\end{array}
\]
Exercise (8.22 from textbook):

Design a Turing machine accepting the following language:

\[ \{ w \in \{0,1\}^* \mid w \text{ has an equal number of } 0's \text{ and } 1's \} \]

solution

The idea is that the head of our TM \( M \) moves back and forth on the tape, "deleting" one 0 for each 1; if there are no 0's and 1's in the end, the string is accepted.

When in state \( q_1 \), \( M \) has found a 1 and looks for a 0; in state \( q_2 \) is the other way around.

Note that the head never moves left of any \( X \), so that there are never unmatched 0's and 1's on the left of an \( X \).

From initial state \( q_0 \), \( M \) picks up a 0 or a 1 and turns it into \( X \). The only final state is \( q_4 \). In state \( q_3 \), \( M \) moves head left looking for the rightmost \( X \).

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>( \delta )</th>
<th>( X )</th>
<th>( Y )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( q_0 )</td>
<td>( q_2, X, R )</td>
<td>( q_3, X, R )</td>
<td>( q_4, \xi, R )</td>
<td>( q_0, Y, R )</td>
<td></td>
</tr>
<tr>
<td>( q_1 )</td>
<td>( q_3, Y, L )</td>
<td>( q_2, 1, R )</td>
<td>( q_2, Y, R )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_2 )</td>
<td>( q_2, 0, R )</td>
<td>( q_3, Y, L )</td>
<td>( q_2, Y, R )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( q_3 )</td>
<td>( q_3, 0, L )</td>
<td>( q_3, 1, L )</td>
<td>( q_0, X, R )</td>
<td>( q_3, Y, L )</td>
<td></td>
</tr>
<tr>
<td>( q_4 )</td>
<td>( q_4, Y, R )</td>
<td>( q_4, Y, L )</td>
<td>( q_4, Y, R )</td>
<td>( q_4, Y, L )</td>
<td>( q_4, Y, R )</td>
</tr>
</tbody>
</table>
Exercise (8.4.2 from textbook)

Consider the following N TM:

\[ M = ( \{ q_0, q_1, q_2 \}, \{ 0, 1 \}, \{ 0, 1, \overline{0}, \overline{1} \}, q_0, \{ q_2 \} ) \]

with \( \delta \) defined as follows:

\[
\begin{array}{c|ccc}
& 0 & 1 & \overline{1} \\
\hline
q_0 & \{ (q_0, 0, R) \} & \{ (q_2, 0, R) \} & \emptyset \\
q_1 & \{ (q_2, 0, R), (q_1, 0, L) \} & \{ (q_1, 0, R), (q_1, 1, L) \} & \{ (q_2, 1, R) \} \\
q_2 & \emptyset & \emptyset & \emptyset \\
\end{array}
\]

Show the 1D's reached by \( M \) when the input is:

(a) 01
(b) 011

Solution

(a) \[
\begin{array}{c}
q_0 \\
1 & q_0 \\
1 & q_0 \\
1 & q_0 \\
1 & q_0 \\
\end{array}
\]

(b) \[
\begin{array}{c}
q_0 \\
1 & q_0 \\
1 & q_0 \\
1 & q_0 \\
1 & q_0 \\
\end{array}
\]

Note that here we do not branch.
Exercise (8.4.5 from textbook)

Suppose you have a tape with all 0's except a single $\#$, with the head in some (unknown) position.

(a) Write a N TM able to enter into a final state (starting from initial state) by scanning $\#$.

(b) Then, write a deterministic TM doing the same job.

Solution

(a) The TM just needs to guess whether $\#$ is on the left or on the right. We call $q, q_f$ the two states ($q_f$ is final).

$$\delta(q, \#) = \{(q, \#, L), (q, \#, R)\}$$

$$\delta(q, \#) = \{(q_f, \#, R)\}$$

(b) The deterministic TM goes back and forth, examining one more position on the tape on the left, and then one on the right; marked symbols are turned from $\#$ to $\#$.

<table>
<thead>
<tr>
<th></th>
<th>$#$</th>
<th>$#$</th>
<th>$#$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>$(q_1, #, L)$</td>
<td>$(q_0, #, R)$</td>
<td>$(q_2, #, R)$</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$(q_0, #, R)$</td>
<td>$(q_3, #, L)$</td>
<td>$(q_2, #, R)$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

In $q_0$, the TM looks for the next $\#$ to the right, while in $q_1$ it looks for the next one on the left. When a $\#$ is reached, it is turned into $\#$ and the search starts over in the opposite direction.