University of Houston

Homework 2 Solutions

COSC 3340
Introduction to Automata & Computability

Instructor: Rathish Das
TA: Aayush Gupta

Exercise 1

For $\Sigma = \{0, 1\}$, design a Turing machine that accepts the language denoted by the regular expression 00^*.

Solution : Starting at the left end of the input, we read each symbol and check that it is a 0. If it is, we continue by moving right. If we reach a blank without encountering anything but 0, we terminate and accept the string. If the input contains a 1 anywhere, the string is not in $L(00^*)$, and we halt in a nonfinal state. To keep of the computation, two internal states $Q = \{q_0, q_1\}$ and one final state $F = \{q_1\}$ are sufficient. As a transition function we can take $\delta(q_0, 0) = (q_0, 0, R)$,

$$\delta(q_0, \Box) = (q_1, \Box, R)$$

As long as 0 appears under the read-write head, the head will move to the right. If at any time a 1 is read, the machine will halt in the nonfinal state q_0, since $\delta(q_0, 1)$ is undefined.

Exercise 2

For $\Sigma = \{a, b\}$, design a Turing machine that accepts

$$L = \{a^n b^n : n \geq 1\}.$$

Solution : Starting at the leftmost a, we check it off by replacing it with some symbol, say x. We then let the read-write head travel right to find the leftmost b, which in turn is checked off by replacing it with another symbol, say y. After that, we go left again to the leftmost a, replace it with an x, then move to the leftmost b and replace it with y, and so on. Traveling back and forth this way, we match each a with a corresponding b. If after some time no a's or b's remain, then the string must be in L.

Working out the details, we arrive at a complete solution for which $Q = \{q_0, q_1, q_2, q_3, q_3\}, F = \{q_4\}. \Sigma = \{a, b\}, \Gamma = \{a, b, x, y, \Box\}$. The transitions can be broken into several parts. The set $\delta(q_0, a) = (q_1, x, R)$,

Exercise 3

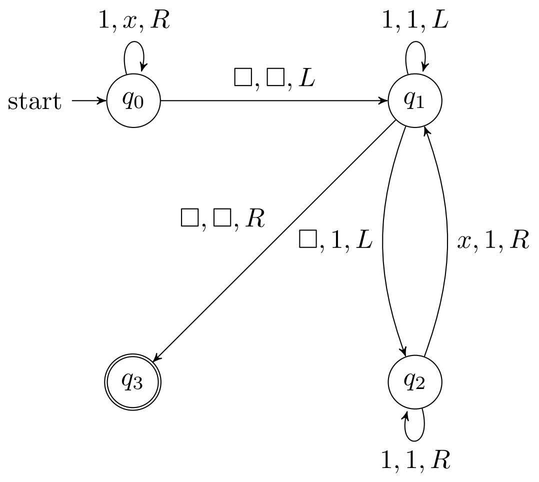
Design a Turing machine that copies strings of 1's. More precisely, find a machine that performs the computation

$$q_0 w \overset{*}{\vdash} q_f ww,$$

for any $w \in \{1\}^+$.

Solution : To solve this problem, we implement the following intuitive process:

1. Replace every 1 by an x.
2. Find the rightmost x and replace it with 1.
3. Travel to the right end of the current nonblank region and create a 1 there.
4. Repeat steps 2 and 3 until there are no more x's.



The solution is shown in the above transition graph. Let us trace the program with the simple string 11. The computation performed in this case is

$$\begin{aligned} q_0 11 &\vdash x q_0 1 \vdash x x q_0 \Box \vdash x q_1 x \\ &\vdash z 1 q_2 \Box \vdash x q_1 11 \vdash q_1 x 11 \\ &\vdash 1 q_2 11 \vdash 11 q_2 1 \vdash 111 q_2 \Box \\ &\vdash 11 q_1 11 \vdash 1 q_1 111 \\ &\vdash q_1 1111 \vdash q_1 \Box 1111 \vdash q_3 1111. \end{aligned}$$

Exercise 4

Give implementation-level descriptions of Turing machines that decide the following languages over the alphabet $\{0, 1\}$.

$$\{w | w \text{ contains an equal numbers of 0s and 1s}\}$$

Solution: " On input string w :

1. Scan the tape and mark the first 0 that has not been marked. If no unmarked 0 is found, go to stage 4. Otherwise, move the head back to the front of the tape.
2. Scan the tape and mark the first 1 which has not been marked. If no unmarked 1 is found, *reject*.
3. Move the head back to the front of the tape and go to stage 1.
4. Move the head back to the front of the tape. Scan the tape to see if any unmarked 1s remain. If none are found, *accept*; otherwise, *reject*."

Exercise 5

Show that the collection of Turing-recognizable languages is closed under the operation of `union`.

Solution : For any two Turing-recognizable languages L_1 and L_2, let M_1 and M_2 be the TMs that recognize them. We construct a TM M' that recognizes the union of L_1 and L_2: " On input w:

1. Run M_1 and M_2 alternatively on w step by step. If either accept, *accept*. If both halt and reject, *reject*"

If either M_1 and M_2 accept w, M' accepts w because the accepting TM arrives to its accepting state after a finite number of steps. Note that if both M_1 and M_2 reject and either of them does so by looping, then M' will loop.