Models of Computation

January 20, 2021

[1]:
```
from IPython.display import Image
```

1 Problem 1 (Sisper 1.1 a-d)

a. What is the start state?

- M_1: q_1
- M_2: q_1

b. What is the set of accept states?

- M_1: q_2
- M_2: q_1, q_4

c. What sequence of states does the machine go through on input **aabb**?

- M_1: q_1, q_2, q_3, q_1, q_1
- M_2: q_1, q_1, q_1, q_2, q_4

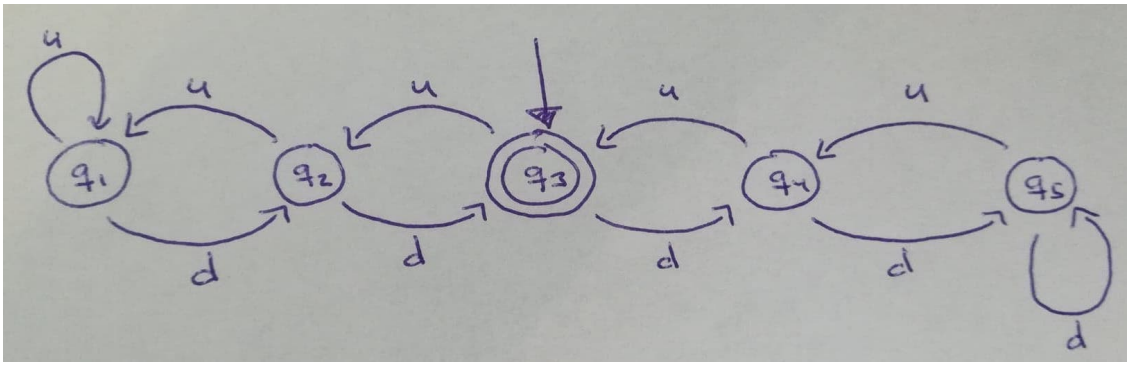
d. Does the machine accept the string **aabb**?

- M_1: No
- M_2: Yes

2 Problem 2 (Sipser 1.3)

[2]:
```
Image(filename = "Sipser 1-3 DFA.jpg", width = 600)
```

[2]:



3 Problem 3 (Sipser 1.31)

For any string $w = w_1, w_2, ..., w_n$, the ***reverse*** of w, written w^R, is the string w om reverse order, $w_n, ..., w_2, w_1$. For any language A, let $A^R = \{w^R | w \in A\}$. Show that if A is regular, so is A^R. (see page 63 in Sipser)

Help provided by https://www.youtube.com/watch?v=7OzQ8ItYGSw

The way to prove this is to take the end state(s) of the DFA (or NFA) and start there with the reversed language. Additionally, we need t reverse all the transitions. This will lead us to end at the start state of the original DFA, and show that the language A^R is regular.

Consider the following DFA designed to accept any strings containing 001:

[3]:
```
Image(filename = "Figure 1-22 Sipser.png", width = 600)
```

[3]:

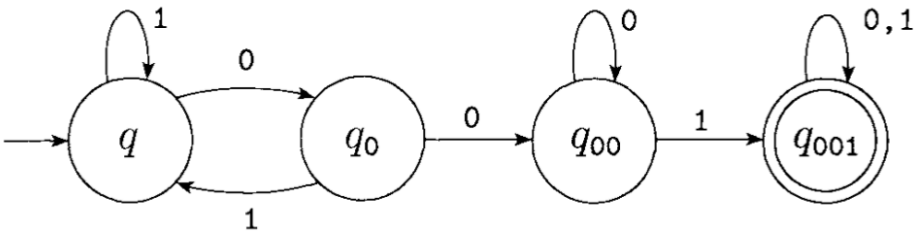
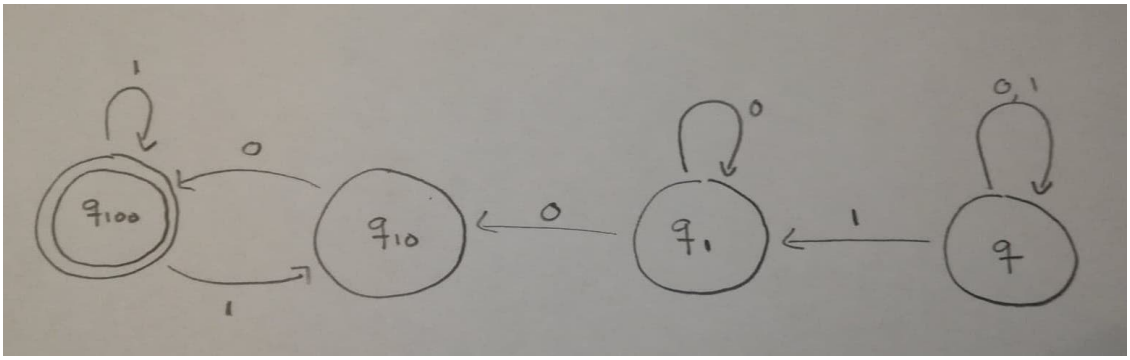


FIGURE 1.22
Accepts strings containing 001

Let's try reversing everything and starting at the original end state to see if the reverse (any string containing 100) is accepted.

[4]:
```
Image(filename = "Reversed DFA.jpg", width = 600)
```

[4]:



Now if we start at q and input any string with 100 in it, we will end up in the accept state. There are now multiple paths that can be taken and it looks like the reverse of the original DFA is actually an NFA, but there is a path that leads to the accept state for any string containing 100, and once the accept state is reached there is no way out because inputing a 1 has a path leading straight back to the accept state and there is no path for an input of 0 once you've reached the accept state.

4 Problem 4 (Sipser 3.2)

[5]:
```
Image(filename = "Example 3-9.png", width = 600)
```

[5]:

EXAMPLE 3.9

The following is a formal description of $M_1 = (Q, \Sigma, \Gamma, \delta, q_1, q_{\text{accept}}, q_{\text{reject}})$, the Turing machine that we informally described (page 139) for deciding the language $B = \{w\#w | \ w \in \{0,1\}^*\}$.

- $Q = \{q_1, \ldots, q_{14}, q_{\text{accept}}, q_{\text{reject}}\}$,
- $\Sigma = \{0,1,\#\}$, and $\Gamma = \{0,1,\#,x,\sqcup\}$.
- We describe δ with a state diagram (see the following figure).
- The start, accept, and reject states are q_1, q_{accept}, and q_{reject}.

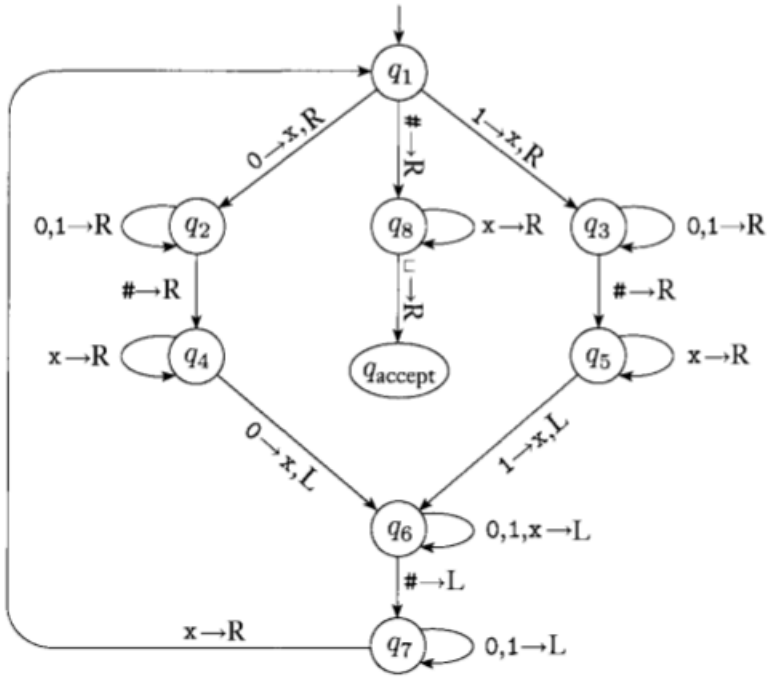


FIGURE 3.10
State diagram for Turing machine M_1

In each of the parts, give the sequence of configuratioins that M_1 enters when started on the indicated input string:

a. 11

- $q_1, 1, 1$ (replace with x, move right)
- $x, q_3, 1$ (move right)
- $x, 1, q_3, blank$ (no path defined for input $blank$)

- Reject

b. 1#1

- $q_1, 1, \#, 1$ (replace with an x and move right)
- $x, q_3, \#, 1$ (move right)
- $x, \#, q_5, 1$ (replace with an x and move left)
- $x, q_6, \#, x$ (move left)
- $q_7, x, \#, x$ (move right)
- $x, q_1, \#, x$ (move right)
- $x, \#, q_8, x$ (move right)
- $x, \#, x, q_8, blank$ (move right)
- Accept

c. 1##1

- $q_1, 1, \#, \#, 1$ (replace with x, move right)
- $x, q_3, \#, \#, 1$ (move right)
- $x, \#, q_5, \#, 1$ (no path specified for input #)
- Loop

d. 10#11

- $q_1, 1, 0, \#, 1, 1$ (replace with x, move right)
- $x, q_3, 0, \#, 1, 1$ (move right)
- $x, 0, q_3, \#, 1, 1$ (move right)
- $x, 0, \#, q_5, 1, 1$ (replace with x, move left)
- $x, 0, q_6, \#, x, 1$ (move left)
- $x, q_7, 0, \#, x, 1$ (move left)
- $q_7, x, 0, \#, x, 1$ (move right)
- $x, q_1, 0, \#, x, 1$ (replace with x, move right)
- $x, x, q_2, \#, x, 1$ (move right)
- $x, x, \#, q_4, x, 1$ (move right)
- $x, x, \#, x, q_4, 1$ (no path defined for input 1)
- Loop

e. 10#10

- $q_1, 1, 0, \#, 1, 0$ (replace with x, move right)
- $x, q_3, 0, \#, 1, 0$ (move right)
- $x, 0, q_3, \#, 1, 0$ (move right)
- $x, 0, \#, q_5, 1, 0$ (replace with x, move left)
- $x, 0, q_6, \#, x, 0$ (move left)
- $x, q_7, 0, \#, x, 0$ (move left)
- $q_7, x, 0, \#, x, 0$ (move right)
- $x, q_1, 0, \#, x, 0$ (replace with x, move right)
- $x, x, q_2, \#, x, 0$ (move right)
- $x, x, \#, q_4, x, 0$ (move right)
- $x, x, \#, x, q_4, 0$ (replace with x, move left)
- $x, x, \#, q_6, x, x$ (move left)
- $x, x, q_6, \#, x, x$ (move left)
- $x, q_7, x, \#, x, x$ (move right)

- $x, x, q_1, \#, x, x$ (move right)
- $x, x, \#, q_8, x, x$ (move right)
- $x, x, \#, x, q_8, x$ (move right)
- $x, x, \#, x, x, q_8, blank$ (move right)
- Accept

(Looks for answers after question section in textbook)

5 Problem 5 (Sipser 3.5)

Examing the formal definition of a Turing machine to answer the following questions, and explain your reasoning.

1. Can a Turing machine ever write the blank symol on its tape?
   - No, because the blank symbol is not in the input alphabet.
2. Can the tape alphabet Γ be the same as the input alphabet Σ?
   - No, because the tape alphabet does contain the blank symbol. The input alphabet is a subset of the tape alphabet.
3. Can a Turing machine's head *ever* be in the same location in two successive steps?
   - Yes, if asked to move left off the tape, the Turing machine will remain in the same place between steps.
4. Can a Turing machine contain just a single state?
   - No, because the machine must have $\{q_{acccept}, q_{reject} | q_{accept} \neq q_{reject}\}$