Lecture #10: Introduction to Turing Machines
Lecture Presentation

Let $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$. The first part of the presentation concerns the Turing machine

$$M_1 = (Q_1, \Sigma, \Gamma_1, \delta_1, q_0, q_{\text{accept}}, q_{\text{reject}})$$

where $\Gamma_1 = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{X}, \sqcup\}$, with an incomplete transition diagram shown in Figure 1 on page 2. This is a Turing machine with a set of states

$$Q_1 = \{q_0, q_{\mathtt{a},1}, q_{\mathtt{a},2}, q_{\mathtt{b},1}, q_{\mathtt{b},2}, q_{\mathtt{c}}, q_{L,1}, q_{L,2}, q_{\text{accept}}, q_{\text{reject}}\}$$

— where the accepting state q_{accept} is shown as "q_A" in the picture, instead. As the above 7-tuple states, this Turing machine has start state q_0, accepting state q_{accept}, and rejecting state q_{reject}. Transitions for states $q \in Q_1 \setminus \{q_A, q_R\}$ and symbols $\sigma \in \Gamma$ that are not shown all have the form $\delta_1(q, \sigma) = (q_R, \sigma, \mathtt{R})$.

Transition Table for This Turing Machine:

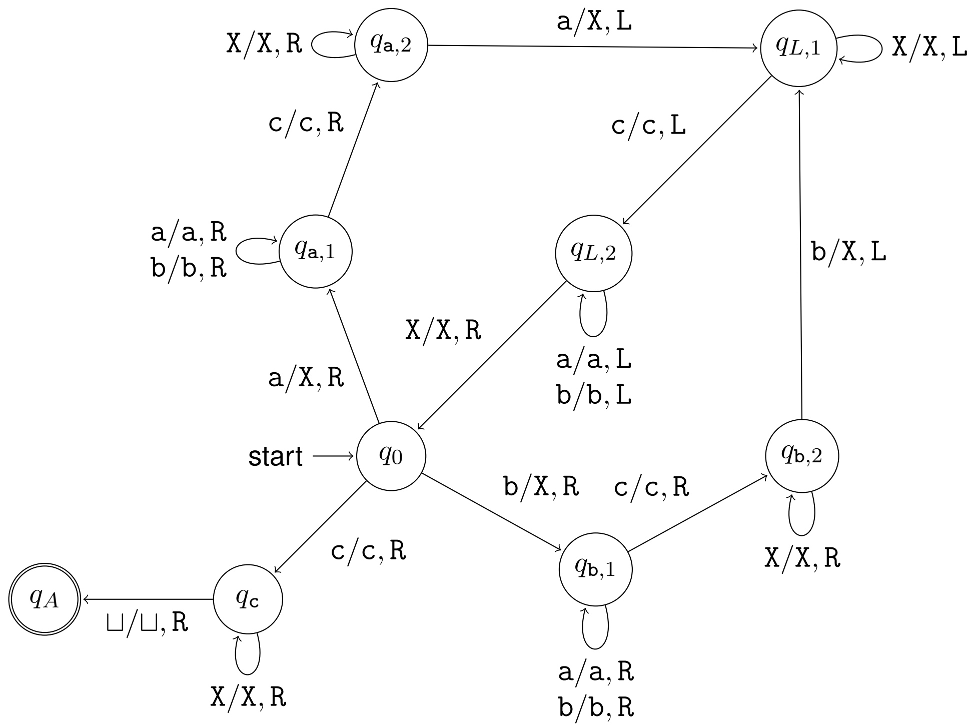

Figure 1: A Turing Machine That Decides a Language

Sequence of configurations for an execution for this Turing machine on the input string

$$\omega = \mathtt{abcab}$$

Now let $\Sigma_1 = \Sigma_2 = \{0, 1\}$. Recall that the ***unpadded binary representation*** of the number 0 is the string 0 with length one in $\Sigma_1^\star$ and that, if n is a positive integer, then the ***unpadded binary representation*** of n is the string

$$\sigma_k\sigma_{k-1}\ldots\sigma_1\sigma_0 \in \Sigma_1^\star$$

with length k, for a non-negative integer k, such that $k \geq 0$, $\sigma_k = 1$, and (if we equate the symbol 1 with the number 1 and if we equate the symbol 0 with the number 0)

$$\sum_{h=0}^{k} \sigma_h \cdot 2^h = n.$$

Thus the unpadded binary representations of the numbers 1, 2, 3 and 4 are the strings 1, 10, 11, and 100, respectively.

Let $L_{\text{bin}} \subseteq \Sigma_1^\star$ be the set of strings in $\Sigma_1^\star$ that are unpadded binary representations of non-negative integers. Then L_{bin} is the union of the set (of size one) $\{0\}$ and the set of non-empty strings in $\Sigma_1^\star$ that begin with 1. Consider a total function $f_{+1} : \Sigma_1^\star \to \Sigma_2^\star$ such that, for all $\omega \in \Sigma_1^\star$,

- If $\omega \in L_{\text{bin}}$, so that ω is the unpadded binary representation of a non-negative integer n, then $f_{+1}(\omega)$ is the unpadded binary representation of the integer $n + 1$.
- If $\omega \notin L_{\text{bin}}$ then $f_{+1}(\omega) = \lambda$, the empty string.

This next part of the presentation concerns the Turing machine

$$M_2 = (Q_2, \Sigma_1, \Sigma_2, \Gamma_2.\delta_2, q_0, q_{\text{halt}})$$

where $\Gamma_2 = \{0, 1, \mathtt{X}, \sqcup\}$, with an incomplete transition diagram shown in Figure 2 on page 4. This is a Turing machine where $\Gamma = \{0, 1, \mathtt{X}, \sqcup\}$ and with a set of states

$$Q_2 = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9, q_{\text{halt}}\}.$$

The halt state, q_{halt}, and transitions leading to it, are not shown in Figure 2. The missing transitions, leading to the halting state, are as follows.

(a) $\delta_2(q_0, \sigma) = (q_{\text{halt}}, \sqcup, \mathtt{L})$ for $\sigma = \sqcup$ and $\sigma = \mathtt{X}$.

(b) $\delta_2(q_1, \mathtt{X}) = (q_{\text{halt}}, \sqcup, \mathtt{L})$ — so that $\delta_2(q_1 \mathtt{X}) = \delta_2(q_1, \sqcup)$.

(c) $\delta_2(q_2, \sigma) = (q_{\text{halt}}, 1, \mathtt{L})$ for $\sigma \in \Gamma$.

(d) $\delta_2(q_3, \mathtt{X}) = (q_{\text{halt}}, \sqcup, \mathtt{L})$ — so that $\delta_2(q_3, \mathtt{X}) = \delta_2(q_3, \sqcup)$.

(e) $\delta_2(q_4, \sigma) = (q_{\text{halt}}, \sqcup, \mathtt{L})$ for $\sigma = \sqcup$ and $\sigma = \mathtt{X}$.

(f) $\delta_2(q_6, \mathtt{X}) = (q_{\text{halt}}, \sqcup, \mathtt{L})$ — so that $\delta_2(q_6, \mathtt{X}) = \delta_2(q_6, \sqcup)$.

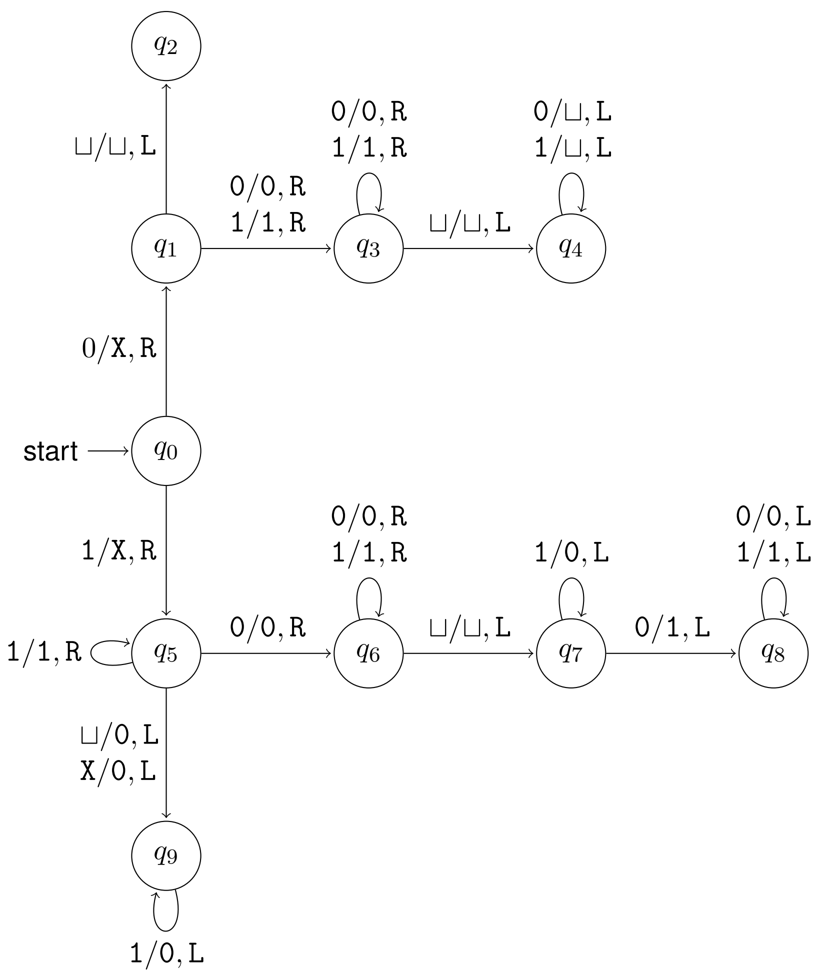

Figure 2: Incomplete State Diagram for a Turing Machine That Computes a Function

(g) $\delta_2(q_7, \sigma) = (q_{\text{halt}}, \sqcup, \mathtt{L})$ for $\sigma = \sqcup$ and $\sigma = \mathtt{X}$.

(h) $\delta_2(q_8, \sigma) = (q_{\text{halt}}, \mathtt{1}, \mathtt{L})$ for $\sigma = \sqcup$ and $\sigma = \mathtt{X}$.

(i) $\delta_2(q_9, \sigma) = (q_{\text{halt}}, \mathtt{1}, \mathtt{L})$ for $\sigma = \sqcup$ and $\sigma = \mathtt{X}$.

It is possible to show that this Turing machine computes a total function $f_M : \Sigma_1^\star \to \Sigma_2^\star$, that is, a function $f_M : \{\mathtt{0}, \mathtt{1}\}^\star \to \{\mathtt{0}, \mathtt{1}\}^\star$.

Sequence of configurations for an execution of this Turing machine on input $\omega = \lambda$:

Conclusion: $f_M(\lambda) =$

Sequence of configurations for an execution of this Turing machine on input $\omega = 0$:

Conclusion: $f_M(0) =$

Sequence of configurations for an execution of this Turing machine on input $\omega = 1$:

Conclusion: $f_M(1) =$

Sequence of configurations for an execution of this Turing machine on input $\omega = 10$:

Conclusion: $f_M(10) =$

Sequence of configurations for an execution of this Turing machine on input $\omega = 11$:

Conclusion: $f_M(11) =$

Suppose that ω is the unpadded binary representation of $2^k - 1$ for a positive integer k, so that ω is a sequence of k copies of "1".

Description of Execution of M_2 on Input ω:

Conclusion: $f_M(\omega)$ is the unpadded binary representation of the integer...

Claim: $f_M = f_{+1}$. That is, M_2 computes the function $f_{+1} : \Sigma_1^\star \to \Sigma_2^\star$ that is given above.

How Can You Prove This?

Consider the execution of M_2 on an input string $\omega \in \Sigma_1^\star$.

Case: $\omega = \lambda$:

Case: $\omega = 0$:

Case: $\omega = 1$:

Case: ω begins with 0 and had length at least two, so that $\omega = 0\nu$ for a string

$$\nu = \alpha_1\alpha_2\ldots\alpha_k$$

for some integer $k \geq 1$ and for $\alpha_1, \alpha_2, \ldots, \alpha_k \in \{0, 1\}$.

Case: ω begins with 1, has length at least two, and does not include any copies of 1 — so that $\omega = 1^k$ for an integer $k \geq 2$.

Case: ω begins with 1, has length at least two, and *does* include at least one copy of 1 — so that $\omega = 1^k 0 \mu$ for some integer $k \geq 1$ and for some string $\mu \in \{0, 1\}^\star$.

Since all cases have been considered , we may now conclude that M_2 computes the function f_{+1}, as claimed.

Note:

- There was a lot to do here, even though this is a rather small and simple Turing machine!
- As the functions to be computed (or the languages to be recognized or decided) get more complicated, it can be easy to get overwhelmed by everything that must be done in order to develop a Turing machine that solves a given problem, and prove it to be correct.
- A supplementary document includes a design technique called ***refinement*** that can help to manage the complexity of problems concerning Turing machines. This will also be discussed in the next lecture.