3.2 This exercise concerns TM M_1, whose description and state diagram appear in Example 3.9. In each of the parts, give the sequence of configurations that M_1 enters when started on the indicated input string.

a. 11.

Solution.

$q_1 11 \to \mathrm{x}q_3 1 \to \mathrm{x}1q_3\sqcup \to \mathrm{x}1\sqcup q_{reject}$

■

b. 1#1.

Solution.

$q_1 1\#1 \to \mathrm{x}q_3\#1 \to \mathrm{x}\#q_5 1 \to \mathrm{x}q_6\#\mathrm{x} \to q_7\mathrm{x}\#\mathrm{x} \to \mathrm{x}q_1\#\mathrm{x} \to \mathrm{x}\#q_8\mathrm{x} \to \mathrm{x}\#q_8\sqcup \to \mathrm{x}\#\mathrm{x}\sqcup q_{accept}\sqcup$

■

c. 1##1.

Solution.

$q_1 1\#\#1 \to \mathrm{x}\ q_3\#\#1 \to \mathrm{x}\#\ q_5\#1 \to \mathrm{x}\#\#\ q_{reject}1$

■

d. 10#11.

Solution.

$q_1 10\#11 \to \mathrm{x}q_1 0\#11 \to \mathrm{x}0q_1\#11 \to \mathrm{x}0\#q_5 11 \to \mathrm{x}0q_6\#\mathrm{x}1 \to \mathrm{x}q_7 0\#\mathrm{x}1 \to q_7\mathrm{x}0\#\mathrm{x}1 \to \mathrm{x}q_1 0\#\mathrm{x}1 \to \mathrm{xx}q_2\#\mathrm{x}1 \to \mathrm{xx}\#q_4\mathrm{x}1 \to \mathrm{xx}\#\mathrm{x}q_4 1 \to \mathrm{xx}\#\mathrm{x}1q_{reject}\sqcup$

■

e. 10#10.

Solution.

$q_1 10\#10 \rightarrow xq_3 0\#10 \rightarrow x0q_3\#10 \rightarrow x0\#q_5 10 \rightarrow x0q_6\#x0 \rightarrow xq_7 0\#x0 \rightarrow q_7 x0\#x0 \rightarrow xq_1 0\#x0 \rightarrow xxq_2\#x0 \rightarrow xx\#q_4 x0 \rightarrow xx\#xq_4 0 \rightarrow xx\#q_6 xx \rightarrow xxq_6\#xx \rightarrow xq_7 x\#xx \rightarrow xxq_1\#xx \rightarrow xx\#q_8 xx \rightarrow xx\#xq_8 x \rightarrow xx\#xxq_8 \sqcup \rightarrow xx\#xx\sqcup q_{accept}\sqcup$

■

3.7 Explain why the following is not a description of a legitimate Turing machine.

M_{bad} = "On input $< p >$, a polynomial over variables $x_1, ..., x_k$:

1. Try all possible settings of $x_1, ..., x_k$ to integers values.
2. evaluate p on all of these settings.
3. If any of these settings evaluates to 0, *accept*; otherwise, *reject*"

Solution.

The above description is not a legitimate Turing machine because the first step requires the Turing machine to store a infinite set of possible settings. The second step of the machine also requires infinite processing time and the machine will never terminate on the third step because the last statement will never be true unless the machine is limited to a finite set of settings.

■

3.11 A Turing machine with doubly infinite tape is similar to an ordinary Turing machine, but its tape is infinite to the left as well as to the right. The tape is initially filled with blanks except for the portion that contains the input. Computation is defined as usual except that the head never encounters an end to the tape as it moves leftward. Show that this type of Turing machine recognizes the class of Turing-recognizable languages.

Solution.

Let TM D be a Turing machine with doubly infinite tape and TM S be an ordinary Turing machine. To show that a TM D can recognize the class of Turing-recognizable languages, we need to show that 1) any language L that can be recognized by TM S can be recognized by a TM D and 2) any language L that can be recognized by a TM D can be recognized by TM S. By marking the left-hand end of the input to detect and prevent the head from moving off of that end, a TM D can simulate TM S. To simulate a TM D by TM S, the first tape of the doubly infinite tape is written with the input string, and the second tape is blank. By cutting the tape of TM D into two parts, the portion with the input string and all the blank spaces to its right appears on the first tape of the doubly infinite tape.

■

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

a. 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: Run M_1 and M_2 alternately on w step by step. If either accepts, accept. If both halt and reject, reject.

■

b. concatenation.

Solution.

Let K and L be two Turing-recognizable languages, and let M_k and M_L denote the Turing machines that recognize K and L respectively. We construct a non-deterministic Turing machine M_{KL} that recognizes the language KL.

Let s be a string from language KL and M_{KL} works for an input string s: Non-deterministically cut input w into w_1 and w_2. Run M_k on w_1, if it halts and rejects, reject. Run M_L on w_2, if it accepts, accept. If it halts and rejects, reject.

■

c. star.

Solution.

For a turing recognizable language L, we construct a non-deterministic Turing machine M_L that recognizes L^*.

On input w: non-deterministically cut w into parts w_1, w_2, ..., w_n. Run M_L on w_i for all i, if M_L accepts all of them, accept. If M_L halts and rejects for any i, reject.

■

d. intersection.

Solution.

Let K and L be two Turing recognizable languages, and let M_K, M_L, $M_{K \cap L}$ denote the Turing machines recognizing K, L, and $K \cap L$ respectively. We use M_K and M_L to construct $M_{K \cap L}$.

On input w: run M_K on w. If it halts and rejects, reject. If it accepts, then run M_L on w. If it halts and rejects, reject. If it accepts, accept.

■

e. homomorphism.

Solution.

Let X a Turing recognizable language be recognized by Turing machine M_x. To recognize h(x), the other Turing machine M_Y is simulated such that:

On input s, it will consider all strings w such that h(w) = s.

The Turing machine M_X will execute on input w by going through all strings in w. If h(w) = s, start executing on M_x on input w. If it accepts, accept. Else S will be rejected.

■

4.2 Consider the problem of determining whether a DFA and a regular expression are equivalent. Express this problem as a language and show that it is decidable.

Solution.

$C = \{< M, R > | \text{ M is a DFA and R is a regular expression with } L(M) = L(R)\}$.

The Turing machine decides C:

On input $< M, R >$, where M is a DFA and R is a regular expression: 1. Convert R into a DFA D_R using the algorithm in the proof of Kleene's theorem. 2. Run TM decider F from theorem 4.5 on input $< M, D_R >$ 3. If F accepts,accept. If F rejects, reject.

■

4.7 Let B be the set of all infinite sequences over {0, 1}. Show that B is uncountable using a proof by diagonalization.

Solution.

Each element in B is an infinite sequence $(b_1, b_2, ...)$, where each $b_i \in \{0, 1\}$. Suppose B is countable, Then by defining correspondences between N and B. Specifically, for n ∈ N, let f(n) = $(b_{n1}, b_{n2}, ...)$ where b_{ni} is the i-th bit in the n-th sequence. Now define the infinite sequence c = $(c_1, c_2, ...) \in B$, where the i-th bit in c is the opposite of the i-th bit in the i-th sequence. Building this shows that c does not equal to any f(n) for any n, which is a contraction. Hence, B is uncountable.

■

4.13 Let A = {< R, S > | R and S are regular expressions and L(R) ⊆ L(S)}. Show that A is decidable.

Solution.

We will design a TM T that decides A: T = On input {< R, S >} where R and S are regular expressions: Construct a DFA B such that $L(B) = \overline{L(S)} \cap L(R)$. Run E_{DFA} on input B. Output what E_{DFA} outputs. Since E_{DFA} is decidable, A is decidable.

■

4.20 Let A and B be two disjoint languages. Say that language C separates A and B if $A \subseteq C$ and $B \subseteq \overline{C}$. Show that any two disjoint co-Turing-recognizable languages are separable by some decidable language.

Solution.

Suppose that A and B are disjoint co Turing recognizable languages. Since A is co Turing recognizable, its complement $\overline{A}$ must have an enumerator. Similarly, the fact that B is co Turing recognizable implies $\overline{B}$ has an enumerator. Since A and B are disjoint, we have that $\overline{A} \cup \overline{B} = \sum^*$. Thus, every string in $\sum^*$ is the union of $\overline{A}$ and $\overline{B}$. Furthermore, since A and B are disjoint, every string in B is in $\overline{A}$ and every string in A is in $\overline{B}$. Let C be the language recognized by the Turing machine, since the Turing machine accepts all strings in A and rejects all strings in B, its language C separates A and B.

■