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.

$q_1 11 \to \mathrm{x} q_3 1 \to \mathrm{x1} q_3 \sqcup \to \mathrm{x1}\sqcup q_{reject}$

■

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.

$q_1 10\#11 \to \mathrm{x} q_1 0\#11 \to \mathrm{x0} q_1 \#11 \to \mathrm{x0}\# q_5 11 \to \mathrm{x0} q_6 \#\mathrm{x1} \to \mathrm{x} q_7 0\#\mathrm{x1} \to q_7 \mathrm{x0}\#\mathrm{x1} \to \mathrm{x} q_1 0\#\mathrm{x1} \to \mathrm{xx} q_2 \#\mathrm{x1} \to \mathrm{xx}\# q_4 \mathrm{x1} \to \mathrm{xx}\#\mathrm{x} q_4 1 \to \mathrm{xx}\#\mathrm{x1} q_{reject} \sqcup$

■

e. 10#10.

Solution.

$q_1 10\#10 \rightarrow \mathrm{x}q_3 0\#10 \rightarrow \mathrm{x}0q_3\#10 \rightarrow \mathrm{x}0\#q_5 10 \rightarrow \mathrm{x}0q_6\#\mathrm{x}0 \rightarrow \mathrm{x}q_7 0\#\mathrm{x}0 \rightarrow q_7\mathrm{x}0\#\mathrm{x}0 \rightarrow \mathrm{x}q_1 0\#\mathrm{x}0 \rightarrow \mathrm{xx}q_2\#\mathrm{x}0 \rightarrow \mathrm{xx}\#q_4\mathrm{x}0 \rightarrow \mathrm{xx}\#\mathrm{x}q_4 0 \rightarrow \mathrm{xx}\#q_6\mathrm{xx} \rightarrow \mathrm{xx}q_6\#\mathrm{xx} \rightarrow \mathrm{x}q_7\mathrm{x}\#\mathrm{xx} \rightarrow \mathrm{xx}q_1\#\mathrm{xx} \rightarrow \mathrm{xx}\#q_8\mathrm{xx} \rightarrow \mathrm{xx}\#\mathrm{x}q_8\mathrm{x} \rightarrow \mathrm{xx}\#\mathrm{xx}q_8\sqcup \rightarrow \mathrm{xx}\#\mathrm{xx}\sqcup q_{accept}\sqcup$

■

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.

■

3.16 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.

■

4.7 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 = $\{< R, S > \mid$ R and S are regular expressions and $L(R) \subseteq L(S)\}$. 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.

■

5.1 Show that EQ_{CFG} is undecidable.

Solution.

Assume the contrary that EQ_{CFG} is decidable. Assume that we have a decider R for EQ_{CFG}. Using R, we construct a decider S for ALL_{CFG} on grammar G with an alphabet of {0, 1}

1. Create a grammar $G_1 = S \rightarrow 0 \mid 1 \mid \epsilon \mid 0S \mid 1S$

2. Run R on $< G, G_1 >$
3. If R accepts, then S accepts. If R rejects, then S rejects.

Since R can tell us if two grammars are equivalent, it determines if G produces all strings by comparing git to grammar G_1 which produces all strings. But since ALL_{CFG} is undecidable, the only mistake was assuming that EQ_{CFG} was decidable. Thus, EQ_{CFG} is undecidable.

■

5.4 If $A \leq_m B$ and B is a regular language, does that imply that A is a regular language? Why or why not?

Solution.

This does not imply that A is a regular language. To show this, we can create a language such that language A is not regular but language B is regular.
Assume that language A is:

$$A = \{ab \mid n \leq 0\} \text{ and } B = \{b\} \text{ over the input } \sum = \{a, b\}$$

$$f(w) = \begin{cases} b & \text{if } w \in A \\ a & \text{if } w \notin A \end{cases} \tag{1}$$

Here, language B is a regular language because it is finite but language is not regular. ■

5.17 Show that the Post Correspondence Problem is decidable over the unary alphabet $\sum = \{1\}$.

Solution.

If the alphabet is unary, then the only difference between the dominoes is the number of 1s that each has on the top and bottom. Consider a Turing Machine M that:

1. If some domino has the same number of 1s on the top and bottom, accept.
2. If some domino have more 1s on top than on bottom, reject. If some domino have less 1s on top than on bottom, reject.
3. Find one domino with more 1s on the top than bottom and one domino with more 1s on bottom than on top. Choosing b of the first domino and a of the second will make an equal number of 1s on both top and bottom, hence a match.

■

5.24 Let J = {w | either w = 0x for some x ∈ A_{TM}, or w = 1y for some y ∈ $\overline{A_{TM}}$}. Show that neither J nor $\overline{J}$ is Turing-recognizable.

Solution.

Let A be the language {$< m, x >$ | M is a TM and M does not accept x}. A is not Turing-recognizable by reduction from $\overline{A_{TM}}$. Firstly, reduce A to J by reducing f(w) = 1w, so that w is not in A if and only if f(w) is in J. The function f is computable and A is not Turing-recognizable, J is not Turing-recognizable. Next we reduce $\overline{A_{TM}}$ to $\overline{J}$ by reducing g(w) to 0w, so that w is in $\overline{A_{TM}}$ if and only if g(w) is in $\overline{J}$. Both functions f and g are computable, since A and $\overline{A_{TM}}$ are not Turing-recognizable, J and $\overline{J}$ are not Turing-recognizable.

■

5.31 Let

$$f(x) = \begin{cases} 3x+1 & \text{for odd x} \\ x/2 & \text{for even x} \end{cases} \tag{2}$$

for any natural number x. If you start with an integer x and iterate f, you obtain a sequence, x, f(x), f(f(X)), ... Stop if you ever hit 1. For example, if x = 17, you get the sequence 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1. Extensive computer tests have shown that every starting point between 1 and large positive integer gives a sequence that ends in 1. But the question of whether all positive starting points end up at 1 is unsolved; it is called the 3x + 1 problem.
Suppose that A_{TM} were decidable by a TM H. Use H to describe a TM that is guaranteed to state the answer to the 3x + 1 problem.

Solution.

If A_{TM} were decidable by a Turing Machine H, then we could create a Turing Machine F that takes a number x and iterates through the f(x) sequence, and stops once it reaches 1. Now by creating a new Turing Machine G that starts with a given input, say i, that calls halting program H on F and i. If H says that F does not halt on i, then stop. If H says that F does halt i, then increase i by 1, and repeat the process. Finally, by calling H on G and 1, if H says that G with 1 will not halt, then there is no counter example to the Collatz conjecture, if H says that G with 1 does halt, there is a counter example to the Collatz conjecture.

■