Homework 8 Solution hints

December 2018

1. (a) \( L = \{ w w^R w \} \)
   1) compute and store \( i = \frac{|\text{input}|}{3} \). if \( i \) is not an integer, reject
   2) use \( i \) to rewrite the input as \( axbxc \) where \( a, b, c \) are words such that \( |a| = |b| = |c| = i \) and \( x \) is some special symbol not in the input
   3) check if the input has form \( axa^R xc \) for \( |a| = i \)
   4) check if the input has form \( cxaxa^R \) for \( |a| = i \)
   5) if (3) and (4) hold, accept
   6) otherwise, reject

(b) \( L = \{ w_1 w_2 | w_1 \neq w_2, |w_1| = |w_2| \} \)
   1) check if the length of input is positive and even, reject if not
   2) store the length of input \( i \)
   3) using \( i \), rewrite the input as \( axb \) where \( a, b \) are words such that \( |a| = |b| = i \) and \( x \) is some special symbol not in the input
   4) check if the first half of input and second half of input differ, accept if yes, reject otherwise

(c) the complement of (b)

   use the solution for (b), but flip accepting/rejecting states

(d) \( L = \{ a^n b^m | n = m^2, m \geq 1 \} \)
   1) check that the input has the form \( a^+ b^+ \), reject if not
   2) for each \( b \) symbol, for each \( b \) symbol, remove one \( a \) symbol.
   3) accept if all \( a \)'s are removed precisely when the \( b \) symbols are exhausted, reject otherwise

(e) \( L = \{ a^n | n \text{ is a prime number} \} \)
   1) reject if the input is \( \lambda \) or \( a \), accept if the input is \( aa \)
   2) store \( i = 2 \)
   3) check if the input is a multiple of \( i \)
   4) if yes, reject
   5) otherwise, increment \( i \)
   6) repeat steps (3)-(6) until \( i + 1 = |\text{input}| \)
   7) accept
2. Write a Turing Machine program that adds three non-zero integers in unary.

Hint: replace the two 0’s in the input with 1’s, then replace the two rightmost 1 symbols with blanks

3. Decoding \( \langle M \rangle \) yields the transition function:
   \[
   \delta(q_1, a_1) = (q_1, a_2, R) \\
   \delta(q_1, a_2) = (q_2, a_2, L)
   \]

(a) Construct the sequence of IDs of \( M \) with the initial ID \( q_1a_1a_1a_1a_2 \).

\[
q_1a_1a_1a_2 \downarrow a_2q_1a_1a_2 \downarrow a_2a_2q_1a_2 \downarrow a_2a_2q_2a_2a_2
\]

Does \( M \) accept the string \( a_1a_1a_2 \)? Yes, because \( q_2 \in F \)

(b) Construct the sequence of IDs of \( M \) with the initial ID \( q_1a_1a_1a_1a_1 \).

\[
q_1a_1a_1a_1 \downarrow a_2q_1a_1a_1 \downarrow a_2a_2q_1a_1 \downarrow a_2a_2a_2q_1a_1 \downarrow a_2a_2a_2a_2q_1
\]

Does \( M \) accept the string \( a_1a_1a_2 \)? No, because \( q_1 \notin F \)

(c) Does \( M \) always halt? Yes.

(d) Determine the language accepted by \( M \).

\[ a_1^*a_2\Sigma^* \text{ or } \Sigma^*a_2\Sigma^* \text{, i.e. all strings containing } a_2 \]

4. Let \( M \) be a Turing Machine with no Left transitions. Show \( L(M) \) is regular.

Let \( M = (Q, \Sigma, \Gamma, \delta, q_0, F) \). We will construct an NFA \( N = (Q', \Sigma, \delta', q_0', F') \) that simulates all computations of \( M \). Because \( M \) cannot move to the left, \( M \) will move to the right with each transition. Thus, the only behavior of \( M \) that \( N \) must simulate is the change is in \( M \)'s state; the symbol \( M \) writes to the tape will never be read by \( M \), thus it need not be remembered by \( N \).

Let \( N \) have the same states as \( M \), i.e. \( Q' = Q \), and let \( N \) have the transition \( \delta'(q_i, a_j) = (q_k) \) whenever \( M \) has a transition \( \delta(q_i, a_j) = (q_k, a_1, \text{Left}) \). Let \( N \) have starting state \( q_0 \) and accepting states \( F \). Note that \( N \)'s behavior is identical to \( M \)'s behavior on all inputs. Thus, there is an NFA that accepts \( L(M) \), so \( L(M) \) is regular.

5. Write a formal and complete Turing Machine program that adds 1 to a nonzero binary number.
Hint: Move to the right end of the input (to the least significant bit). If the least significant bit is a 0, flip it to 1 and halt. Otherwise, travel left on each 1 and flip it to 0. Upon seeing a 0, flip this bit to 1 and halt. If the input is a string of 1’s (no 0 is seen before the left end is reached), add a 1 to the left of the input and halt.

6. Suppose $M$ is a Turing Machine that recognizes $L$. Are the following true or false?

(a) $M$ halts on every $w \in L$
   true, by the definition of recognizes

(b) If $L$ is decidable, then $M$ always halts.
   false. $M$ recognizes $L$; $M$ may not halt on words not in $L$. Assuming $L$ is decidable means there exists some machine that always halts and recognizes $L$; it need not be the case that $M$ is this machine.

(c) If $M$ always halts, then $L$ is decidable.
   true. A language is decidable if it is recognized by a TM that always halts. Assuming $M$ always halts and given that $M$ recognizes $L$, $L$ is decidable.

(d) If $L$ is nonempty, then $M$ halts on some $w$.
   true. $M$ is a recognizer for $L$, thus $M$ halts on all words in $L$. Given that $L$ is nonempty, there is some $w'$ in $L$, thus $M$ halts on $w'$.

(e) There is a TM that recognizes the language $L^C$.
   false. $L$ may be recognizable, yet undecidable. In this case, no such TM exists.

(f) There is a TA that recognizes the language $L^*$.  
   true, the class of recognizable languages is closed under Kleene closure.

7. Show that the family of recursively enumerable languages is closed under union.

A language $L$ is recursively enumerable (r.e.) if there is some TM that recognizes $L$. Thus, for each member $L_i$ of the family of r.e. languages, there is a machine $M_i$ that recognizes $L_i$. Let $L$ be the union of $n$ r.e. languages: $L = L_1 \cup \ldots \cup L_n$. Let $M$ be a TM such that on input $w$, $M$ simultaneously simulates machines $M_1, \ldots, M_n$ on $w$ and halts in an accepting state if any of these machines halts in an accepting state. Let $w$ be an arbitrary word in $L$. By the definition of $L$, $w \in L(M_i)$ for some $1 \leq i \leq n$, thus machine $M_i$ will halt in an accepting state on input $w$, thus $M$ will halt in an accepting state on input $w$. Thus, $M$ recognizes $L$, and $L$ is in the family of recursively enumerable languages.

8. Show how to simulate a standard Turing Machine that always halts by a Turing Machine that always leaves its tape blank before it halts.
Hint: Let $M$ be a standard Turing Machine. Construct a TM $M'$ that first marks the left and right ends of the input with a special symbol like #, then simulates $M$, using the # symbols to track the leftmost and rightmost cells visited. When $M$ enters a halting state, $M'$ travels to the right # symbol on the tape, changes this symbol to a blank symbol, then travels to the left # symbol while replacing all non-blank symbols on the tape with the blank symbol, then halts.