PROBLEMS

Part 1: These 4 problems are to be turned in and each will be graded (10 points each).

1. Describe a TM which accepts the language
   \[ L = \{ w \text{ } 2 w^R | w \text{ is a string in } \{0,1\}^* \text{ and } w^R \text{ is the string } w \text{ in reverse order } \}. \]

   Here I mean informally describe how the TM works. You need not give the full program or diagram.

   We are looking for something like the description of the TM in examples 3.11 or 3.12 of the book on pages 174 and 175.

   Answer:
   Basic idea of
   The TM T to accept the language L above (in fact this T will decide L):
   On an input string x from the input alphabet \{0,1,2\},
   1. T begins by marking off the leftmost input symbol (using a symbol to replace a 0, say by a, and a 1, say b). It then uses a state to remember which of 0 or 1 was the leftmost symbol.
   2. T then checks the input string to see if it has the right form to be in L. Specifically it checks if the string consist of a string of binary bits followed by a “2” followed by another string of binary bits followed by a B. If not, then T rejects, otherwise it then moves its head back to the leftmost square of the tape.
   3. T then enters a loop which uses different states to move up and back through the string and one-by-one marks off a symbol from the w part of the input string with the corresponding string \(w^R\) in the right half. It it find a mismatch of left and right symbols in rejects, otherwise it keeps going.
   4. Eventually all the w string will be matched up with symbols from the \(w^R\) string. If either of the 2 halves runs out of symbols before the other, T rejects. If not it ends up at the middle “2” having marked off each of the bits from either side of the “2”. At this point T accepts its input.

2. (i). Write a TM program which accepts the set of binary string with an even number of 1’s, rejects all binary string which have exactly one 1 and loops on all other binary strings.

   Answer: The TM uses only 0, 1 and B on its tape and has 6 states.
Idea: State 0 means TM has seen 0 1’s so far. State 1 means TM has seen 1 1’s so far. State 2 means TM has seen an even number 1’s so far. State 3 means TM has seen an odd number \( \geq 3 \) of 1’s so far.

The TM has the following program \( d \).

\[
\begin{align*}
d(0,0) &= (0,0,R) \\
d(0,1) &= (1,1,R) \\
d(1,0) &= (1,0,R) \\
d(1,1) &= (2,1,R) \\
d(2,0) &= (2,0,R) \\
d(2,1) &= (3,0,R) \\
d(3,0) &= (3,0,R) \\
d(3,1) &= (2,0,R) \\
d(0,B) &= (a,0,R) \\
d(1,B) &= (r,B,R) \\
d(2,B) &= (a,0,R) \\
d(3,B) &= (3,B,R)
\end{align*}
\]

(ii). Write a TM program which decides the set \{ a, ba, acb \}. Your input alphabet should be \{ a, b, c \}.

Answer: The TM uses only a, b, c and B on its tape and has 7 states.

\[
\begin{align*}
d(0,a) &= (1,a,R) \\
d(0,b) &= (2,b,R) \\
d(0,c) &= (r,c,R) \\
d(1,a) &= (r,a,R) \\
d(1,b) &= (r,b,R) \\
d(1,c) &= (3,c,R) \\
d(2,a) &= (4,a,R) \\
d(2,b) &= (r,a,R) \\
d(2,c) &= (r,c,R) \\
d(3,a) &= (r,a,R) \\
d(3,b) &= (4,b,R) \\
d(3,c) &= (r,c,R) \\
(4,a) &= (r,a,R) \\
(4,b) &= (r,b,R) \\
(4,c) &= (r,c,R) \\
d(4,B) &= (a,B,R) \\
d(1,B) &= (a,B,R) \\
d(2,B) &= (a,a,R)
\end{align*}
\]


Answer: As long as you can state clearly what the regular TM does that the new type of TM defined here cannot do, then you are fine.

For example, you might say that the new machine cannot decide the language \( L \) from problem 1 above. By 1 a standard TM can be found that does decide \( L \). It cannot be decided by the new machine as a TM cannot just read an arbitrary string \( w \) from left to right and remember all of it. In order to do this the machine has to compare long \( w \)'s with the \( w^R \) symbols character by character and so must go left and right over \( w \) repeatedly which the more limited machine is restricted from doing.

4. Problem 3.16, parts b and d.

Answer:

Part b. Show that given 2 recognizable languages \( J \) and \( L \) we can find a recognizer for \( JL \) = the concatenation of \( J \) and \( L \).

Answer: Let \( T1 \) be a TM which recognizes \( J \) and \( T2 \) be a TM which recognizes \( L \).

A TM \( T3 \) to recognize \( JL \) runs as follows.

1. On input \( w \) to \( T3 \) where \( |w| = n \) and \( w = w_1w_2...w_n \), we consider \( n+1 \) possible pairs of strings \( s_i, t_i \) for \( i = 1,2,...,n+1 \) which could be concatenated to equal \( w \) and which need to be checked to see if \( s_i \in J \) and \( t_i \in L \).
These $n$ pairs are given by $s_i = w_1 w_2 ... w_i, t_i = w_{i+1} w_{i+2} ... w_n$. (Note that when $i = n$ we let $t_n$ be the empty string, and when $i = n+1$ we let $s_{n+1}$ be the empty string and $t_{n+1} = w_1 w_2 ... w_n$.

T3(w) begins by writing all $n+1$ of these pairs $s_i, t_i$ on its tape and checking for each if $s_i$ is a string from T1’s alphabet and if $t_i$ is a string from T2’s alphabet. If not it erases pair. $s_i, t_i$

2. Now the pairs that remain on T3’s tape after step 1 are the pairs that T3 will check to see if $s_i \in J$ and $t_i \in L$.

T3(w) now carries out this checking by interleaving the $2(n+1)$ computations of $T1(s_i)$ and $T2(t_i)$ for all of the remaining pairs on T3’s tape. T3 keeps track of which of these computations of T1 and T2 accept and T3(w) accepts if and only if it eventually finds pair $s_i, t_i$ for one of the i’s for which both $T1(s_i)$ and $T2(t_i)$ accept.

Part d. Show that given 2 recognizable languages $J$ and $L$ we can find a recognizer for $J \cap L = \text{the intersection of } J \text{ and } L$.

Answer:
Let T1 be a TM which recognizes J and T2 be a TM which recognizes L.

A TM T3 to recognize $J \cap L$ runs as follows.

1. On an input w, T3 first runs T1(w).
   - If T1(w) ever halts and accepts then T3(w) goes to step 2.
   - If T1(w) ever halts and accepts then T3(w) reject.

2. Now T3 runs T2(w) and it does whatever T2(w) does.

Note that TW3(w) accepts if and only if both T1(w) accepts and T2(w) accepts, which happens if and only if $w \in (J \cap L)$.

Part 2: These problems are good practice and you should try them. They will not be graded

1. Exercise 3.1, parts b and c
2. Exercise 3.2, parts c and d
3. Exercise 3.5, all parts
4. Exercise 3.7
5. Problem 3.22