CMSC 28100-1 / MATH 28100-1
Introduction to Complexity Theory
Fall 2017 – Bonus Homework
Solution

November 17, 2017

Exercise 1 (HMU 8.4.1)**.** Informally but clearly describe multitape Turing machines that accept each of the following languages from HW3, Exercise 1. Try to make your Turing machines run in time $O(n)$.

(a) The set of strings with an equal number of 0's and 1's.

(b) $\{a^n b^n c^n : n \geqslant 1\}$.

(c) $\{ww^R : w$ is any string of 0's and 1's$\}$, where w^R is the reverse of a string. For instance, we have $10010^R = 01001$.

Solution. We start with item (a).

The idea is to keep a signed unary counter on the second tape (the sign is kept in a state) and keep track of the difference between the number of 0's and 1's read.

The transitions are given in Table 1.

For item (b), we use a unary counter on the second tape to count the a's and use it to check if the number of b's and c's is the same.

The transitions are given by

	q_0	q_1	q_2	q_f
(a,a)	$-$	$-$	$-$	$-$
(a,b)	$-$	$-$	$-$	$-$
(a,c)	$-$	$-$	$-$	$-$
(a,B)	(q_0,a,a,R,R)	$-$	$-$	$-$
(b,a)	$-$	(q_1,b,a,R,L)	$-$	$-$
(b,b)	$-$	$-$	$-$	$-$
(b,c)	$-$	$-$	$-$	$-$
(b,B)	(q_1,b,B,S,L)	$-$	$-$	$-$
(c,a)	$-$	$-$	(q_2,c,a,R,R)	$-$
(c,b)	$-$	$-$	$-$	$-$
(c,c)	$-$	$-$	$-$	$-$
(c,B)	$-$	(q_2,c,B,S,R)	$-$	$-$
(B,a)	$-$	$-$	$-$	$-$
(B,b)	$-$	$-$	$-$	$-$
(B,c)	$-$	$-$	$-$	$-$
(B,B)	$-$	$-$	(q_f,B,B,R,R)	

where q_0 is the initial state and q_f is the

For item (c), the idea is to copy the input to the second tape then compare the input with the reverse of the second tape. We also need to ensure that the word has even length.

The transitions are given in Table (c).

	$(0,0)$	$(0,1)$	$(0,B)$	$(1,0)$	$(1,1)$	$(1,B)$	$(B,0)$	$(B,1)$	(B,B)
q_0	$-$	$-$	$(q_+,0,1,R,R)$	$-$	$-$	$(q_-,1,1,R,R)$	$-$	$-$	(q_f,B,B,R,R)
q_+	$-$	$-$	$(q_+,0,1,R,R)$	$-$	$-$	$(p_+,1,B,S,L)$	$-$	$-$	$-$
p_+	$-$	$-$	$-$	$-$	$(q_+,1,B,R,S)$	$(q_0,1,B,R,R)$	$-$	$-$	$-$
q_-	$-$	$-$	$(p_-,0,B,S,L)$	$-$	$-$	$(q_-,1,1,R,R)$	$-$	$-$	$-$
p_-	$-$	$(q_-,0,B,R,S)$	$(q_0,0,B,R,R)$	$-$	$-$	$-$	$-$	$-$	$-$
q_f	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$-$

Table 1: Transitions for item (a). In the above q_0 is the initial state and q_f is the accepting state.

	$(0,0)$	$(0,1)$	$(0,B)$	$(1,0)$	$(1,1)$	$(1,B)$	$(B,0)$	$(B,1)$	(B,B)
q_0	$-$	$-$	$(q_1,0,0,R,R)$	$-$	$-$	$(q_1,1,1,R,R)$	$-$	$-$	(q_2,B,B,L,S)
q_1	$-$	$-$	$(q_0,0,0,R,R)$	$-$	$-$	$(q_0,1,1,R,R)$	$-$	$-$	$-$
q_2	$-$	$-$	$(q_2,0,B,L,S)$	$-$	$-$	$(q_2,1,B,L,S)$	$-$	$-$	(q_3,B,B,R,L)
q_3	$(q_3,0,0,R,L)$	$-$	$-$	$(q_3,1,1,R,L)$	$-$	$-$	$-$	(q_f,B,B,R,L)	
q_f	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$-$

Table 2: Transitions for item (c). In the above q_0 is the initial state and q_f is the accepting state.

$\triangleleft$

Exercise 2. Design (single tape) Turing machines for the following languages.

(a) $L = \{w \in \{0,1\}^* : w$ contains more zeroes than ones$\}$.

(b) $L = \{w \in \{0,1\}^* : w$ contains exactly twice as many zeroes as ones$\}$.

Solution. For item (a), we essentially do the same as in Exercise 1a of HW3.

The transition function is given by

	0	1	B	$*$
q_0	$(q_1, *, R)$	$(q_2, *, R)$	(q_f, B, L)	$(q_0, *, R)$
q_1	$(q_1, 0, R)$	$(q_3, *, L)$	(q_f, B, L)	$(q_1, *, R)$
q_2	$(q_3, *, L)$	$(q_2, 1, R)$	$-$	$(q_2, *, R)$
q_3	$(q_3, 0, L)$	$(q_3, 1, L)$	(q_0, B, R)	$(q_3, *, L)$
q_f	$-$	$-$	$-$	$-$

where q_0 is the initial state and q_f is the accepting state.

For item (b), the idea is also similar to the one in Exercise 1a of HW3.

	0	1	B	$*$
q_s	$(q_0, *, R)$	$(q_1, *, R)$	(q_f, B, L)	$(q_0, *, R)$
q_0	$(q_0, 0, R)$	$(q_{01}, *, R)$	$-$	$(q_0, *, R)$
q_1	$(q_{01}, *, R)$	$(q_{11}, *, R)$	$-$	$(q_1, *, R)$
q_{01}	$(q_{01}, 0, R)$	$(q_r, *, L)$	$-$	$(q_{01}, *, R)$
q_{11}	$(q_r, *, L)$	$(q_{11}, 1, R)$	$-$	$(q_{11}, *, R)$
q_r	$(q_r, 0, L)$	$(q_r, 1, L)$	(q_s, B, R)	$(q_r, *, L)$
q_f	$-$	$-$	$-$	$-$

where q_s is the initial state and q_f is the accepting state. $\triangleleft$

Exercise 3 (HMU 8.3.2)**.** A common operation in Turing machine programs involves "shifting over": ideally, we would like to create an extra cell at the current head position, in which we could store some character. However, we cannot edit the tape in this way. Rather, we need to move the contents of each of the cells to the right of the current head position one cell right, and then go our way back to the current head position. Show how to perform this operation. Hint: leave a special symbol to mark the position to which the head must return.

Solution. For each $\sigma \in \Gamma$, we introduce a new symbol $*$ and a new state q_σ. We also introduce three new states q_s, q_r and q_f. The subroutine is activated by transitioning to q_s with the head on the space in which one wants to create an extra cell and is finished in state q_f with the head on the new cell created (which will have a blank symbol). We also assume that the calling machine never writes blank symbols (this can be achieved by replacing all blank symbols written with a new symbol $\widetilde{B}$ that represents it).

The transitions of the subroutine are given by

$$\begin{aligned}
\forall \sigma \in \Gamma, \delta(q_s, \sigma) &= (q_\sigma, *, R);\\
\forall \sigma, \tau \in \Gamma \setminus \{B\}, \delta(q_\sigma, \tau, R) &= (q_\tau, \sigma, R);\\
\delta(q_\sigma, B) &= (q_r, \sigma, L);\\
\forall \sigma \in \Gamma, \delta(q_r, \sigma) &= (q_r, \sigma, L);\\
\delta(q_r, *) &= (q_f, B, S).
\end{aligned}$$

$\triangleleft$

Exercise 4. Give a sufficiently detailed description of how to construct a (multitape) Turing machine which computes the exponentiation function $f(n) = 2^n$. You can assume the input n is given in unary form 1^n, and your Turing machine should produce a unary representation of the number 2^n (i.e., your machine has to produce 1^{2^n}), and halt.

Solution. The algorithm is as follows.

We first move 1^n to the second tape (so that the first tape becomes blank). Then we write a 1 on the first tape. Then we consecutively write a new 1 on the first tape and subtract 1 from the second tape (treating it as a binary number). Since 1^n is the binary number representation of the number $2^n - 1$, this algorithm correctly ends with 1^{2^n} on the first tape. $\triangleleft$