University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
Z. Parsaeian

Theoretical Computer Science - Bridging Course
Sample Solution Exercise Sheet 5

Due: Tuesday, 28th of May 2023, 12:00 pm

Exercise 1: Operation Shift *(3+3 Points)*

Consider a Turing machine $\mathcal{M}$ that is given an arbitrary input string over alphabet $\Sigma = \{1, 2, \dots, n\}$ on its input tape. We would like $\mathcal{M}$ to insert an empty cell, i.e., $\sqcup$, at the beginning of the tape without removing any symbol on the tape. As an example, the Turing machine is supposed to change the input tape of the form $\langle 2, 4, 4, 6, 1, 8, 4, \sqcup, \sqcup, \dots \rangle$ to $\langle \sqcup, 2, 4, 4, 6, 1, 8, 4, \sqcup, \sqcup, \dots \rangle$. Although this operation is not explicitly defined for a Turing machine, one can consider such an operation as shifting the whole string one cell to the right on the input tape.

(a) Give a formal definition of $\mathcal{M}$ to perform the desired operation such that $\mathcal{M}$ recognizes the language Σ^*.

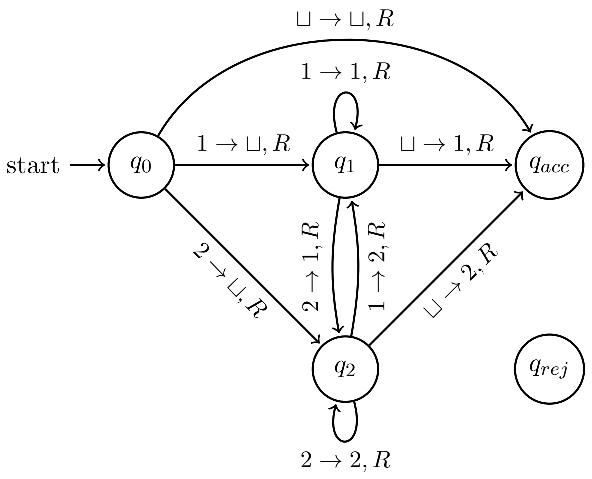
(b) For $n = 2$, i.e., $\Sigma = \{1, 2\}$, draw the state diagram of your constructed Turing machine.

Sample Solution

(a) Consider the set of states to be $Q = \{q_0, q_{acc}, q_{rej}\} \cup \{q_c | c \in \Sigma\}$. Then, the transition functions are as follows.

$$\begin{aligned}
&\forall c \in \Sigma;\ \delta(q_0, c) = (q_c, \sqcup, R)\\
&\forall c, c' \in \Sigma;\ \delta(q_c, c') = (q_{c'}, c, R)\\
&\forall c \in \Sigma;\ \delta(q_c, \sqcup) = (q_{acc}, c, R)\\
&\delta(q_0, \sqcup) = (q_{acc}, \sqcup, R)
\end{aligned}$$

(b)



Remark: whenever needed in the upcoming exerices, we can detect the beginning of the input on the tape e.g. by initially shifting all the input one step to the right (cf. exercise 1). Hence, we can always have a blank symbol placed at the leftmost of the tape.
Alternatively, one can use a trickier method mentioned from the book by Sipser: *"A trickier method of finding the left-hand end of the tape takes advantage of the way that we defined the Turing machine model. Recall that if the machine tries to move its head beyond the left-hand end of the tape, it stays in the same place. We can use this feature to make a left-hand end detector. To detect whether the head is sitting on the left-hand end, the machine can write a special symbol over the current position while recording the symbol that it replaced in the control. Then it can attempt to move the head to the left. If it is still over the special symbol, the leftward move didn't succeed, and thus **the head must have been at the left-hand end**. If instead it is over a different symbol, some symbols remained to the left of that position on the tape. Before going farther, the machine must be sure to restore the changed symbol to the original."*

Exercise 2: Constructing TMs (Part 1) *(3+3 Points)*

1. Consider alphabet $A = \{1, 2, \ldots, 9\}$. We call a string S over A a *blue* string, if and only if the string consisting of the odd-positioned symbols in S is the reverse of the string consisting of the even-positioned symbols in S. For example $S = 14233241$ is a blue string since the substring of the odd-positioned symbols is 1234 which is the reverse of the substring of the even-positioned symbols, i.e., 4321.

   Design a Turing machine which accepts all blue strings over A. You do not need to provide a formal description of the Turing machine but your description has to be detailed enough to explain every possible step of a computation.

2. Construct a Turing machine that decides on the languages
   $C_1 = \{a^i b^j c^k | i - j = k \text{ and } i, j, k \geq 1\}$ and $C_2 = \{a^i b^j c^k | i \times j = k \text{ and } i, j, k \geq 1\}$.

Sample Solution

1. On input S, first go through all symbols. If it is an odd number, reject. Else repeat the following:

   Go left until you reach the first unmarked symbol, mark it, go right to the last unmarked symbol, mark it, and compare both symbols. If they are different, reject.

   Accept if all symbols are marked.

2. For both languages, we first scan the string from left to right to verify that it is of form $a^+b^+c^+$, if it is scan to start of tape and if not, reject.

   - For C_1 we continue as follows: cross off the first a and scan until the first the first b occurs, then shuttle between the a's and b's crossing off one of each until all b's are gone. If all a's have been crossed off and some b's remain,reject. Else, continue shuttling between the the a's and c's crossing off one of each until all c's are gone. Similarly, if all a's have been crossed off and some c's remain, reject. At the end, if some a's remain, reject; else accept.
   - For C_2 we continue as follows: cross off the first a and scan until the first b occurs. Shuttle between b's and c's crossing off one of each until all b's are gone. If all c's have been crossed off and some b's remain,reject. Then, restore the crossed off b's and repeat the previous step if there are a's remaining (to restore the b''s, we need to distinguish between the location of the b's and c's, hence when shuttling, we can start with the c's from the end of input; alternatively incorporate different cross-off symbols for b and c). If all a's are gone, check if all c's are crossed off; if so, accept; else reject.

Exercise 3: Constructing TMs (Part 2) *(3+2+2+1 Points)*

Let $\Sigma = \{0, 1\}$. For a string $s = s_1 s_2 \dots s_n$ with $s_i \in \Sigma$, let $s^R = s_n s_{n-1} \dots s_1$ be the *reversed* string. *Palindromes* are strings s for which $s = s^R$. Then $L = \{sas^R \mid s \in \Sigma^*, a \in \Sigma \cup \{\varepsilon\}\}$ is the language of all palindromes over Σ.

(a) Give a state diagram of a Turing machine recognizing L.

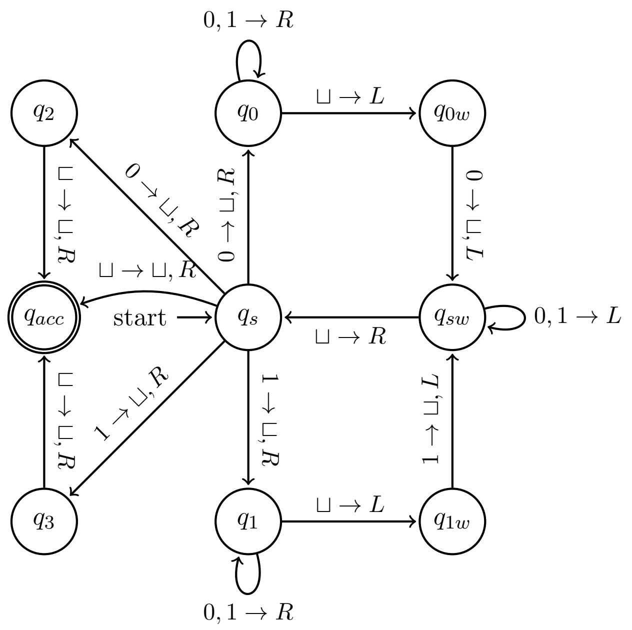
(b) Give the maximum number (or a close upper bound for the number) of head movements your Turing machine makes until it halts, if started with an input string $s \in \Sigma^*$ of length $|s| = n$ on its tape.

(c) Describe (informally) the behavior of a 2-tape Turing machine which recognizes L and uses significantly fewer head movements on long inputs than your 1-tape Turing machine.

(d) Give the maximum number (or a close upper bound for the number) of head movements your Turing machine makes on any of the two tapes until it halts, if started with an input string $s \in \Sigma^*$ of length $|s| = n$ on the first tape.

Sample Solution

(a) *TM description (although not asked for this part):* starting from state q_s, read the first symbol on the input tape and move to state q_0 if it's 0 or state q_1 if it's 1 then replace it with a blank. Move right till the end of the tape (first blank symbol). Move left one symbol. If this symbol is a 0 and you are in state q_{0w}, or if it is 1 and you are in state q_{1w}, replace it with a blank and return all the way to the left until you find a blank symbol, and then move one cell to the right to get back to q_s (otherwise, the word is not a palindrome and you halt-reject by going to state q_{rej}) . Continue in this manner until you halt-reject or all symbols on the tape have been replaced with blanks, in which case you halt-accept by going to state q_{acc}.



Remark: To simplify this figure, we don't show the reject state nor the transitions that are going to the reject state. Those transitions will occur *implicitly* whenever a state lacks an outgoing transition for a particular symbol. For example, from states q_2, q_3, and q_{0w}, there are no outgoing transitions for the symbol 1, then if a 1 occurs under the head when the machine is in these states, it directly goes to q_{rej}.

(b) If n is even, then the head of the Turing machine will move at most $(n+1)+n+(n-1)+\ldots+1 = \frac{(n+1)(n+2)}{2}$ steps (this sum is at most n^2 if n was large enough).
If n is odd, then the head of the Turing machine will move at most $(n+1)+n+(n-1)+\ldots+2 = \frac{(n+1)(n+2)}{2} - 1$ steps (which is also at most n^2 for large enough n).
To see this we can e.g. argue by induction on n and prove that *for even $n \geq 0$, the tape head moves at most $\sum_{i=1}^{n+1} i$ steps until it halts.* (Note we are saying *at most* here, since the string doesn't necessarly need to be accepted and hence can halt-reject with less number of head steps than this sum). (Another note, we can similarly show by induction that when $n \geq 1$ is odd, we move less by 1 head movements from the sum above).
Indeed, *base case:* for $n = 0$, the head moves only 1 step until it reaches q_{acc} and halts.
Induction hypothesis: for an input string of even length $n \geq 0$, suppose the tape head moves at most $\sum_{i=1}^{n+1} i$ steps until it halts.
Induction step : Consider an input string of even length $n+2 \geq 0$, we want to show that the head moves at most $\sum_{i=1}^{n+3} i$ steps until we halt. Indeed, starting from state q_s, reading the first symbol and replacing it with a blank, until we reach the last symbol on the input string (we would be in either q_{0w} or q_{1w}), this takes at most $n+3$ head movements. Now, going back all the way to the left until we see the first non blank symbol (hence until we reach again state q_s) will take at most $n+2$ extra head movements. At this point, we are back to state q_s where the remaining (all non blank symbols) input substring is of length n and the head is pointing at the first non blank symbol of this substring. We can then use the induction hypoythesis on this remaining substring and see that the machine takes at most $\sum_{i=1}^{n+1} i$ head steps until it halts. Hence, in total the head would have moved at most $\sum_{i=1}^{n+3} i$ steps until it halts as desired, and this ends the proof.

(c) Move the tape head of the input tape to the end, and then read backwards to the start of the input tape (e.g. after shifting the whole string on the input tape one cell to the right cf. exercise 1). As you go, write the tape symbols in order on the second tape so that you end up with the reverse of the input tape on the second tape. Reset both tape heads, and then move each to the end of the tape, at each step comparing the symbols each head is pointing to. If you find a position where the symbols are different, the input is not a palindrome and you halt-reject. If you get to the end (first blank symbol on the end) without finding a mismatch then it is a palindrome and you halt-accept.

(d) The heads move three times through the input on both tapes (disregarding the head moves for the shift operation), leading to $O(n)$ (i.e. an upper bound of some constant times n) head moves in total.