University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
D. Atalay, S. Faour

Theoretical Computer Science - Bridging Course
Sample Solution Exercise Sheet 6

Due: Tuesday, 25th of November 2025, 12:00 pm

Exercise 1: Constructing TMs (continued) *(2+2 Points)*

Construct a Turing machine that decides on the languages:

(a) $C_1 = \{a^i b^j c^k | i - j = k \text{ and } i, j, k \geq 1\}$

(b) $C_2 = \{a^i b^j c^k | i \times j = k \text{ and } i, j, k \geq 1\}$

NB: A high level description is enough.

Sample Solution

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.

(a) 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.

(b) 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 2: A Better TM Variant *(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.

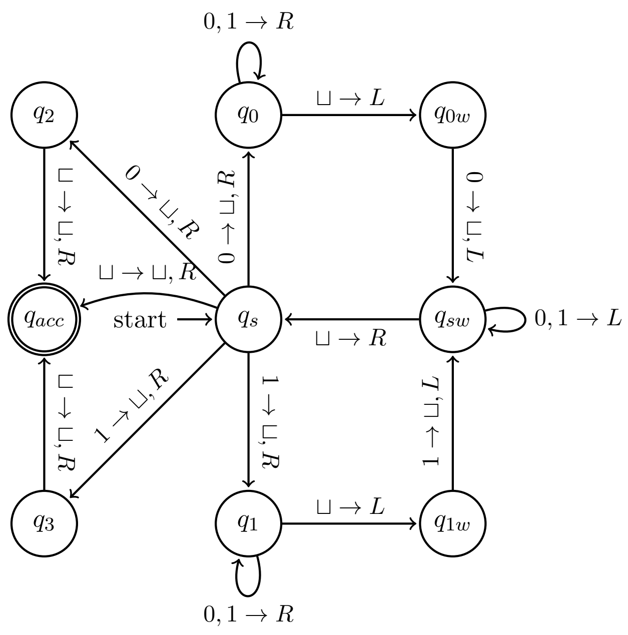
(c) Describe (informally) the behavior of a 2-tape Turing machine which recognizes L and uses significantly fewer head movements on long inputs than your 1-tape Turing machine.

(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

We describe both Turing machines by their behaviour.

(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) The head of the Turing machine whether n is even or odd 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 we choose n large enough for example for $n \geq 6$).

(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.

Exercise 3: Decidable Problems *(3+3+2 Points)*

(a) Show that the following languages are decidable.

- $A = \{\langle R, S\rangle \mid R$ and S are regular expressions and $L(R) \subseteq L(S)\}$.
- $B = \{\langle G\rangle \mid G$ is a CFG over $\{0,1\}$ and $1^* \cap L(G) \neq \phi\}$.
  Hint:Use the fact that a language $C \cap R$ is context free for some context free language C and regular language R.

(b) Consider a decidable language L. Show that its complement $\overline{L}$ is also decidable.

Sample Solution

(a)
- Let T be the Turing Machine deciding the language $\{\langle D\rangle \mid D$ is a DFA with $L(D) = \emptyset\}$ (known from the lecture). We have $L(R) \subseteq L(S) \Leftrightarrow L(R) \setminus L(S) = \emptyset$. Thus we construct a decider for A in the following way.
  A'= " On input $\langle R, S\rangle$ where R, S are regular expression:
  1. Convert R and S into equivalent DFAs (like in the previous lectures)
  2. Construct a DFA D for the regular language $L(R) \setminus L(S) = L(R) \cap \overline{L(S)}$ (we know how to do so from previous lectures).
  3. Run T on input $\langle D\rangle$. Accept iff T accepts."
- From the the fact, we deduce that $1^* \cap L(G)$ is context free. We construct a TM that decides on B as follows.
  B' =" On input $< G >$, where G is a context free grammar :
  1. Construct a CFG C such that $L(C) = 1^* \cap L(G)$ (as described in the tutotrial)
  2. Test whether $L(C) = \phi$ using the decider R for E_{CFG} (cf. lecture).
  3. If R accepts, reject; if R rejects, accept."

(b) As the language L is decidable, it has a Turing Machine M that halts on every input string w. So, it either accepts and halts, or rejects and halts. Using this, we can construct a TM M' that does the following :

1. Run M on input w.
2. If M accepts, M' rejects.
3. If M rejects, M' accepts.

This TM M' not only recognizes the language $\overline{L}$, but also is decidable, as it halts on every input.