Albert-Ludwigs-Universität, Inst. für Informatik
Prof. Dr. Fabian Kuhn
P. Bamberger, Y. Maus,

Theoretical Computer Science - Bridging Course
Summer Term 2017
Exercise Sheet 6

Hand in (electronically or hard copy) by 12:15 pm, December 4th, 2017

Exercise 1: Designing a Turing Machine *(6 Points)*

Design a Turing machine which accepts the language $L = \{w\#\overline{rev}(w) \mid w \in \{0,1\}^*\}$ where $\overline{rev}$ denotes the reverse complement, i.e., $\overline{rev}(a_1a_2,\dots,a_{n-1}a_n) = \overline{a_na_{n-1},\dots,a_2a_1}$ with $\overline{0} = 1$ and $\overline{1} = 0$.

Remark: It is sufficient to give a detailed description of the Turing Machine. You do not need to give a formal definition.

Sample Solution

Proof Sketch: We first check whether the string is of the format $\{0,1\}^*\#\{0,1\}^*$. Then we compute the complement of the left hand string by reading through it and substituting every 0 with a 1 and every 1 with a 0.
Then we do a comparison where we always compare the first symbol of the left string with the last symbol of the right hand string. If they are the same we overwrite the corresponding tape positions with a new symbol Z and continue with the remaining string.

Exersive 2: Semi-Decidable vs. Recursively Enumerable *(5 Points)*

Very often people in computer science use the terms *semi-decidable* and *recursively enumerable* equivalently. The following exercise shows in which way they actually are equivalent. We first recall the definition of both terms.

A language L is *semi-decidable* if there is a Turing machine which accepts every $w \in L$ and does not accept any $w \notin L$ (this means the TM can either reject $w \notin L$ or simply not stop for $w \notin L$.

A language is *recursively enumerable* if there is a Turing machine which eventually outputs every word $w \in L$ and never outputs a word $w \notin L$.

(a) Show that any recursively enumerable language is semi-decidable.

(b) Show that any semi-decidable language is recursively enumberable.

Sample Solution

(a) Let M_L be the TM which enumerates L. Construct a TM which, on input w, simulates M_L. If M_L outputs w the TM accepts w, otherwise it might run forever.

(b) Let M_L be a TM which semi-decides L. We use a tricky simulation of M_L to construct a TM which recursively enumerates L. We order all words lexicographically $w_1, w_2, w_3, \dots$ and then we simulate M_L as follows

1) Simulate one step of M_L on w_1
2) Simulate one (further) step of M_L on w_1 and w_2
3) Simulate one (further) step of M_L on w_1, w_2 and w_3
4) Simulate one (further) step of M_L on w_1, w_2, w_3 and w_4
5) etc.

Exercise 3: Halting Problem *(3+2+2+2 Points)*

The *special halting problem* is defined as

$$H_s = \{\langle M\rangle \mid \langle M\rangle \text{ encodes a TM and } M \text{ halts on } \langle M\rangle\}.$$

(a) Show that H_s is undecidable.

Hint: Assume that M is a TM which decides H_s and then construct a TM which halts iff M does not halt. Use this construction to find a contradiction.

(b) Show that the special halting problem is recursively enumerable.

(c) Show that the complement of the special halting problem is not recursively enumerable.

Hint: What can you say about a language L if L and its complement are recursively enumerable? (if you make some observation for this, also prove it)

(d) Let L_1 and L_2 be recursively enumerable languages. Is $L_1 \setminus L_2$ recursively enumerable as well?

(e) Is $L = \{w \in H_s \mid |w| \leq 1742\}$ decidable? Explain your answer!

Sample Solution

(a) Assume that H is decidable. Then there is a TM M which decides it. Now define a TM $\tilde{M}$ which terminates on the inputs on which M does not terminate: The TM $\tilde{M}$ on input w uses M to test whether $w \in H$. If $w \in H$ it enters a non terminating loop, otherwise it terminates. We now apply $\tilde{M}$ on input $\langle\tilde{M}\rangle$ and construct a contradiction.

$\langle\tilde{M}\rangle \notin H$: Then M rejects $\langle\tilde{M}\rangle$. Thus $\tilde{M}$ terminates on $\langle\tilde{M}\rangle$ by the definition of $\tilde{M}$. Thus $\langle\tilde{M}\rangle \in H$, a contradiction.

$\langle\tilde{M}\rangle \in H$: Then M accepts $\langle\tilde{M}\rangle$, i.e., $\tilde{M}$ enters a non terminating loop on $\langle\tilde{M}\rangle$ and does not halt on $\langle\tilde{M}\rangle$ which means that $\langle\tilde{M}\rangle \notin H$, a contradiction.

(actually both cases are similar as in both cases $\tilde{M}$ enters a non terminating loop and we do have the statement

$$\langle\tilde{M}\rangle \in H \Leftrightarrow \langle\tilde{M}\rangle \notin H.$$

(b) The special halting problem is semi-decidable because we can construct a TM which semi-decides it as follows: If the input is not a valid coding of a TM the TM rejects it. If the input is the coding of a TM M it simulates M on $\langle M\rangle$ and accepts if this simulation stops.

With the previous exercise it follows that the halting problem is recursively enumerable.

(c) First note that if a language L and its complement are recursively enumerable the language L is a recursive language: Assume that L is recursively enumerable by TM M_1 and its complement by TM M_2. Then we construct a TM which, on input w interchangebly simulates one step of M_1 and one step of M_2. Eventually one of the two TMs will output w. If M_1 outputs w we accept w and if M_2 outputs w we reject w.

If the complement of the special halting problem was recursively enumerable, then H and its complement would be recursively enumerable. But then H would be a recursive language which is a contradiction.

(d) This does not hold in general. Let $L_1 = \{0,1\}^*$ be the language of all words over $\Sigma = \{0,1\}$ and let L_2 be the special halting problem. Then L_1 and L_2 are recursively enumerable (L_1 is even a recursive language) but $L_1 \setminus L_2$ equals the complement of the special halting problem and is not recursively enumerable.

(e) Even though we do not know what the language is we know that all words in the language have length at most 1742, that is, the language is finite. So, no matter which words with length of at most 1742 are actually contained in the language there is even a deterministic finite automaton which tests for it, i.e., the language is even regular!