Theoretische Informatik

Prof. Dr. Juraj Hromkovič
Dr. Hans-Joachim Böckenhauer
`https://courses.ite.inf.ethz.ch/theoInf20`

Exemplary Solutions – Sheet 7

Zürich, November 6, 2020

Solution to Exercise 17

We first show using an indirect proof that L_1 is not recursively enumerable. To this end, assume that L_1 is recursively enumerable. Then there exists a Turing machine M such that $L(M) = L_1$. Since M must occur in the canonical enumeration of all Turing machines, there exists some $j \in \mathbb{N}$ such that $M_j = M$. We now consider the word w_{3j+1}. Then we derive

$$\begin{aligned} w_{3j+1} \in L_1 &\iff M_j \text{ does not accept } w_{3j+1} \quad \text{(by definition of } L_1\text{)} \\ &\iff M \text{ does not accept } w_{3j+1} \quad \text{(by definition von } M\text{)} \\ &\iff w_{3j+1} \notin L(M) \\ &\iff w_{3j+1} \notin L_1. \end{aligned}$$

This is a contradiction, our assumption was thus false and L_1 is not recursively enumerable.

An analogous proof does not work for the language L_2. The proof for L_1 is based on the fact that the enumeration contains every Turing machine, in particular one for the language under consideration. This is true for L_2 as well, but does not lead to a contradiction, since this Turing machine can have an index that is not of the form $3i + 1$ for any $i \in \mathbb{N}$.

Note that one cannot conclude that L_2 is recursively enumerable from the failed proof attempt.

Solution to Exercise 18

Let $L \subseteq \{0, 1\}^*$ be an infinite language. Using the diagonalization method, we can define a subset L' of L that is not recursively enumerable. Let M_i be the i-th Turing machine in the canonical order. Then we define a diagonalization language with respect to L as follows:

$$\begin{aligned} L_{\text{diag},L} = \{w \in \{0,1\}^* \mid w = w_i \text{ is the } i\text{-th word in } L \text{ for some } i \in \mathbb{N} \\ \text{and } M_i \text{ does not accept } w_i\}. \end{aligned}$$

It clearly holds that $L_{\text{diag},L} \subseteq L$.

Assume that $L_{\text{diag},L}$ is recursively enumerable. Then $L_{\text{diag},L} = L(M)$ holds for a TM M. Since M must occur in the canonical enumeration of all Turing machines, there exists some $i \in \mathbb{N}$ such that $M = M_i$. But the language $L_{\text{diag},L}$ cannot be equal to $L(M_i)$ because

$$w_i \in L_{\text{diag},L} \iff w_i \notin L(M_i)\,.$$

This is a contradiction, hence, $L_{\text{diag},L} \notin \mathcal{L}_{\text{RE}}$.

Solution to Exercise 19

The path of the frog depends on two parameters that are unknown to us: The origin of its path $u \in \mathbb{Z}$ and the number of steps $s \in \mathbb{Z} - \{0\}$ it makes per night. A negative number of steps denotes a direction of movement to the left. Based on these parameters, the frog is located at the point $u + s \cdot t$ during the day $t \in \mathbb{N}$.

To catch the frog, it suffices to successively try out all pairs of parameters (u, s), i.e., to provide an enumeration of $\mathbb{Z} \times (\mathbb{Z} - \{0\})$. In the lecture (cf. Lemma 5.11 in the textbook), we have seen an enumeration of $(\mathbb{N} - \{0\}) \times (\mathbb{N} - \{0\})$ which assigns the number $f((a,b)) = \binom{a+b-1}{2} + b$ to the pair (a, b). From that, we can derive an enumeration g of $(\mathbb{Z} - \{0\}) \times (\mathbb{Z} - \{0\})$ which assigns the following natural number to the pair $(u, s) \in (\mathbb{Z} - \{0\}) \times (\mathbb{Z} - \{0\})$:

$$g((u,s)) = \begin{cases} 4(\binom{u+s-1}{2} + s) & \text{for } u, s > 0 \\ 4(\binom{u-s-1}{2} - s) - 1 & \text{for } u > 0 \text{ and } s < 0 \\ 4(\binom{-u+s-1}{2} + s) - 2 & \text{for } u < 0 \text{ and } s > 0 \\ 4(\binom{-u-s-1}{2} - s) - 3 & \text{for } u, s < 0 \end{cases}$$

This enumeration uses the enumeration from the lecture for each of the four quadrants and proceeds in turns, enumerating successively one element from each quadrant in one turn. Since f is a bijection, g is clearly a bijection as well.

So far, we have not considered the possibility that the origin of the frog's path can be at the point 0. To provide an enumeration h of $\mathbb{Z} \times (\mathbb{Z} - \{0\})$, we use the enumeration g and alternate between one element from $(\mathbb{Z} - \{0\}) \times (\mathbb{Z} - \{0\})$ (at even positions) and one element from $\{0\} \times (\mathbb{Z} - \{0\})$ (at odd positions). This yields

$$h((u,s)) = \begin{cases} 2g((u,s)) & \text{for } u \neq 0 \\ 4s - 1 & \text{for } u = 0 \text{ and } s > 0 \\ -4s - 3 & \text{for } u = 0 \text{ and } s < 0 \end{cases}$$

At odd positions, the enumeration alternates between a pair $(0, s)$ with $s > 0$ and one with $s < 0$. The function defined in this way is a bijection of the parameter space of the frog's path into the natural numbers.

This yields the following strategy to catch the frog: Test the pair of parameters (u, s) such that $h((u, s)) = t$ on the day $t \in \mathbb{N}$, i.e., try to catch the frog at the point $u + s \cdot t$. This strategy tests eventually each possible value of the parameters, so the frog will be caught in finite time.