ETH zürich
Departement Informatik

Theoretische Informatik

Prof. Dr. Juraj Hromkovič
Dr. Hans-Joachim Böckenhauer
https://courses.ite.inf.ethz.ch/theo_inf_21

Exemplary Solutions – Sheet 6

Zürich, November 5, 2021

Solution to Exercise 16

(a) We assume that the guests are numbered as follows: let $G_{0,i}$ denote the guest who is already accommodated in the room Z_i, for $i \in \mathbb{N}$, and let $G_{j,i}$ denote the i-th guest in the bus j, for $j \in \{1,2,3\}$ and $i \in \mathbb{N}$.

To meet all the guests' requirements, the porter can accommodate the guests as follows. He splits the sequence of rooms into blocks of size 6 and assigns rooms with numbers 0 mod 6 to the present guests, rooms with numbers 1 mod 6 to the guests from the first bus, rooms with numbers 2 mod 6 and 3 mod 6 to guests from the second bus, and rooms with numbers 4 mod 6 and 5 mod 6 to guests from the third bus.

Formally, the room distribution can be described by a function f such that, for all $i \in \mathbb{N}$,

$$\begin{aligned}
&f(G_{0,i}) = Z_{6i}, && \\
&f(G_{1,i}) = Z_{6i+1}, && \\
&f(G_{2,i}) = Z_{6\cdot\frac{i}{2}+2} = Z_{3i+2}, && \text{if } i \text{ is even,}\\
&f(G_{2,i}) = Z_{6\cdot\frac{i-1}{2}+3} = Z_{3i}, && \text{if } i \text{ is odd,}\\
&f(G_{3,i}) = Z_{6\cdot\frac{i}{2}+4} = Z_{3i+4}, && \text{if } i \text{ is even,}\\
&f(G_{3,i}) = Z_{6\cdot\frac{i-1}{2}+5} = Z_{3i+2}, && \text{if } i \text{ is odd.}
\end{aligned}$$

(b) The porter first splits the rooms in infinitely many groups of infinite size each. Then each arriving group of guests (finite or infinite) can be accommodated in the next available group of rooms.

To determine the groups of rooms, we use the following idea: we know that there are infinitely many prime numbers. Let p_i be the i-th prime number in increasing order. Then we define the i-th group of rooms as follows: $\text{Gruppe}(i) = \{p_i^j \mid j \in \mathbb{N}^+\}$. No room can be assigned to multiple groups because every number has a unique prime decomposition.

Solution to Exercise 17

We first show using an indirect proof that L_2 is not recursively enumerable. To this end, suppose that L_2 is recursively enumerable. Then there exists a Turing machine M such that $L(M) = L_2$. Since M must occur in the canonical enumeration of all Turing machines, there exists some $j \in \mathbb{N} - \{0\}$ such that $M_j = M$. Now we consider the word $w_i = w_{2j}$. Then we derive

$$\begin{aligned} w_i = w_{2j} \in L_2 &\iff M_{\lceil i/2 \rceil} = M_j \text{ does not accept } w_{2j} \quad \text{(by definition of } L_2\text{)} \\ &\iff M \text{ does not accept } w_{2j} \quad \text{(by definition von } M\text{)} \\ &\iff w_{2j} \notin L(M) \\ &\iff w_{2j} = w_i \notin L_2. \end{aligned}$$

This is a contradiction, our assumption was thus false and L_2 is not recursively enumerable. An analogous proof does not work for the language L_1. The proof for L_2 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_1 as well, but does not lead to a contradiction, since this Turing machine can have an odd index, which is not of the form $2i$ for any $i \in \mathbb{N} - \{0\}$. Note that one cannot conclude from the failed proof attempt that L_1 is recursively enumerable.

Solution to Exercise 18

(a) For instance, we can define the new model as follows. A Turing machine in the new model is a 7-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$, where

  (i) $Q = \{q_0, q_{\text{accept}}, q_{\text{reject}}\}$,
  (ii) Σ is an alphabet with $¢, \llcorner\!\lrcorner \notin \Sigma$,
  (iii) Γ is an alphabet with $¢, \llcorner\!\lrcorner \in \Gamma$, $\Sigma \subseteq \Gamma$, and $\Gamma \cap Q = \emptyset$,
  (iv) $\delta\colon \{q_0\} \times \Gamma^3 \to Q \times \Gamma^3 \times \{\text{L}, \text{R}, \text{N}\}$ is a transition function such that

$$\forall s, s' \in \Gamma\colon\ \exists q \in Q, \tilde{s}, \tilde{s}' \in \Gamma, d \in \{\text{R}, \text{N}\}\colon \qquad \delta(q_0, (¢, s, s')) = (q, (¢, \tilde{s}, \tilde{s}'), d).$$

We can directly take the definition of a configuration from the standard model without any changes. In contrast, the definition of a step must be adjusted to the new transition function δ:

  (i) $x_1 \dots x_{i-1} q x_i \dots x_n \vdash_M x_1 \dots x_{i-2} y_{\text{L}} p y y_{\text{R}} x_{i+2} \dots x_n$,
  if $\delta(q_0, (x_{i-1}, x_i, x_{i+1})) = (p, (y_{\text{L}}, y, y_{\text{R}}), \text{N})$
  (ii) $x_1 \dots x_{i-1} q x_i \dots x_n \vdash_M x_1 \dots x_{i-2} p y_{\text{L}} y y_{\text{R}} x_{i+2} \dots x_n$,
  if $\delta(q_0, (x_{i-1}, x_i, x_{i+1})) = (p, (y_{\text{L}}, y, y_{\text{R}}), \text{L})$
  (iii) $x_1 \dots x_{i-1} q x_i \dots x_n \vdash_M x_1 \dots x_{i-2} y_{\text{L}} y p y_{\text{R}} x_{i+2} \dots x_n$,
  if $\delta(q_0, (x_{i-1}, x_i, x_{i+1})) = (p, (y_{\text{L}}, y, y_{\text{R}}), \text{R})$, for $i < n - 1$, and
  (iv) $x_1 \dots x_{n-2} q x_{n-1} x_n \vdash_M x_1 \dots x_{n-2} y_{\text{L}} y p y_{\text{R}} \llcorner\!\lrcorner$,
  if $\delta(q_0, (x_{n-2}, x_{n-1}, x_n)) = (p, (y_{\text{L}}, y, y_{\text{R}}), \text{R})$.

We assume that the read/write head is not adjusted on the square with the symbol $¢$, but on the square at the right of it, the initial configuration for an input word x is thus $¢q_0x$ if $|x| \geq 2$, $¢q_0x\llcorner\!\lrcorner$ if $|x| = 1$, and $¢q_0\llcorner\!\lrcorner\llcorner\!\lrcorner$ if $|x| = 0$. The remaining definitions — the notion of a computation and the notion of the language accepted by a Turing machine — can be directly taken from the standard model without any changes.

(b) Let a Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ in the standard model be given. We provide an equivalent Turing machine M' in the new model. The approach is to extend the alphabet Γ by $\Gamma \times Q$ so that the current state of M can be saved in a symbol that can be read again in the next step.

Formally, we define

$$M' = (\{q_0, q_{\text{accept}}, q_{\text{reject}}\}, \Sigma, \Gamma \cup \Gamma \times Q, \delta', q_0, q_{\text{accept}}, q_{\text{reject}}),$$

where, for all $q, \tilde{q} \in Q$, $s, s_{\text{L}}, s_{\text{R}}, \tilde{s} \in \Gamma$, and $S \in \{\text{L}, \text{R}, \text{N}\}$,

$$\delta'(q_0, \big(s_{\text{L}}, s, s_{\text{R}}\big)) = (q_0, \big(s_{\text{L}}, \begin{pmatrix} s \\ q_0 \end{pmatrix}, s_{\text{R}}\big), \text{N}),$$

and, if $\delta(q, s) = (\tilde{q}, \tilde{s}, S)$ with $\tilde{q} \notin \{q_{\text{accept}}, q_{\text{reject}}\}$, then

$$\begin{aligned}
\delta'(q_0, \big(s_{\text{L}}, \begin{pmatrix} s \\ q \end{pmatrix}, s_{\text{R}}\big)) &= (q_0, \big(\begin{pmatrix} s_{\text{L}} \\ \tilde{q} \end{pmatrix}, \tilde{s}, s_{\text{R}}\big), \text{L}), \text{ if } S = \text{L}, \\
\delta'(q_0, \big(s_{\text{L}}, \begin{pmatrix} s \\ q \end{pmatrix}, s_{\text{R}}\big)) &= (q_0, \big(s_{\text{L}}, \tilde{s}, \begin{pmatrix} s_{\text{R}} \\ \tilde{q} \end{pmatrix}\big), \text{R}), \text{ if } S = \text{R}, \text{ and} \\
\delta'(q_0, \big(s_{\text{L}}, \begin{pmatrix} s \\ q \end{pmatrix}, s_{\text{R}}\big)) &= (q_0, \big(s_{\text{L}}, \begin{pmatrix} \tilde{s} \\ \tilde{q} \end{pmatrix}, s_{\text{R}}\big), \text{N}), \text{ if } S = \text{N},
\end{aligned}$$

and, if $\delta(q, s) = (\tilde{q}, \tilde{s}, S)$ with $\tilde{q} \in \{q_{\text{accept}}, q_{\text{reject}}\}$, then

$$\delta'(q_0, \big(s_{\text{L}}, \begin{pmatrix} s \\ q \end{pmatrix}, s_{\text{R}}\big)) = (\tilde{q}, \big(s_{\text{L}}, \tilde{s}, s_{\text{R}}\big), \text{S}).$$