ETH zürich
Departement Informatik

Theoretische Informatik

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

Exemplary Solutions – Sheet 9

Zürich, November 20, 2020

Solution to Exercise 23

To prove $(L_{\text{empty}})^{\complement} \leq_{\text{m}} (L_{\text{diag}})^{\complement}$, we provide a mapping of inputs for $(L_{\text{empty}})^{\complement}$ to inputs for $(L_{\text{diag}})^{\complement}$. Inputs that do not have the form $\text{Kod}(M)$ for any TM M (and are thus contained in $(L_{\text{empty}})^{\complement}$) are mapped to a word $w = w_i$ such that M_i is a TM that accepts every input. Then $w \in (L_{\text{diag}})^{\complement}$.

For every input of the form $\text{Kod}(M)$, we construct a TM M' that ignores its own input and simulates M on all words in parallel as follows: one step on the first word in canonical order, two steps on the first two words, etc. If M accepts some word, then M' finds this accepting computation in finite time. Then M' halts and accepts its own input. This means that, if M accepts some word, then M' accepts every input. If M does not accept any word, then the simulation run by M' continues indefinitely and thus M' does not halt on any input.

Now we compute the index i of M' in canonical order, such that $M' = M_i$ for the i-th Turing machine in the canonical order of all Turing machines, and map the input $\text{Kod}(M)$ for $(L_{\text{empty}})^{\complement}$ to the input w_i for $(L_{\text{diag}})^{\complement}$. Let Σ be the alphabet of M. Then

$$\begin{aligned}
\text{Kod}(M) \in (L_{\text{empty}})^{\complement} &\iff \text{there exists some } w \in L(M) \\
&\iff L(M') = L(M_i) = \Sigma^* \\
&\iff w_i \in L(M_i) \\
&\iff w_i \in (L_{\text{diag}})^{\complement}.
\end{aligned}$$

This concludes the proof of $(L_{\text{empty}})^{\complement} \leq_{\text{m}} (L_{\text{diag}})^{\complement}$.

Solution to Exercise 24

The idea behind the construction of A is to simulate 12 steps of M in 6 steps of A. To this end, every 12 cells of the working tape of M are combined into one cell of A. The same compression is applied to the input as well, A uses its second working tape for that. We note that A can simulate a constant number of computation steps of M in a single step if it has saved the symbols read by M in those steps in its state in advance. We thus pay for optimizing the running time by a significant blow-up of the working alphabet and the set of states.

The MTM A has the same input on its input tape as M. To shorten the computation, A first compresses this input to the second working tape. To this end, it always reads 12 cells of the input tape and writes the 12-tuple of input symbols in one cell of the second working tape. This clearly requires $n + 1$ steps on an input of length n since the head on the input tape reaches the right endmarker \$ at the $(n + 1)$-th step. Afterwards, A moves the head on the second working tape back to the start. Because $\lceil n/12 \rceil$ cells are used on that tape, this requires $\lceil n/12 \rceil$ steps. Hence, A needs $\frac{13n}{12} + c_1$ steps in total for the preprocessing, for a small constant c_1.

The actual simulation of M by A proceeds in *rounds* of up to 6 computation steps. In every round, A simulates 12 computation steps of M. This yields a running time of the simulation at most $\frac{\mathrm{Time}_M(n)}{2} + c_2$, for some small constant c_2.

At the beginning of every round, A reads the contents of its two working tapes at the current cell and the two neighbouring cells at the left and right and saves these $3 \cdot 12 = 36$ cells of the input and working tape of M in its states. This requires 4 steps: one step to the left, two steps to the right, and one step back to the original position. Hence, A now has enough information to simulate 12 steps of M in its states since M can move by at most 12 positions in 12 steps and A knows at least 12 cells to the left and right of the current position of M. In the 12 simulated steps, M can only change the contents of two of the three blocks of 12 cells. These changes can be performed by A in at most 2 steps on its tapes as follows: We only describe the modificaton of the first working tape, changing the head position on the simulated input tape can be performed analogously.

In the fifth step of the round, A changes the content of the current cell of the working tape according to the 12 computation steps of M and moves the head to the left or right if changes are required in the corresponding part of M's tape simulated there. In the sixth step, A performs modifications on the neighbouring cell and potentially returns back. If changes are only necessary on the current position of both working tapes of A, then the sixth step is not needed.

Overall, this yields the claimed upper bound on the total running time of A.

Solution to Exercise 25

Because the given 1-tape-TM always halts, it never reaches the same internal configuration twice during a computation. If it did, then there would be a loop between the two configurations that would repeat infinitely often and thus the TM would not halt. This makes the length of every computation bounded by the number of pairwise distinct internal configurations. The number of internal configurations can be computed as the product of the number of states, the number of possible head positions on the input tape, the number of possible head positions on the working tape, and the number of possible contents of the working tape. The number $|Q|$ of states is constant, the used part of the input tape has length $n + 2$, because it contains just the input word and the two endmarkers, and, by assumption, the used part of the working tape including the left endmarker has length $s(n) + 1$, i.e., there are $s(n) + 1$ possible head positions on the working tape.

To count the internal configurations, it remains to bound the number of possible contents of the working tape. In addition to the left endmarker 0 at the beginning and the fixed infinite suffix $kkk\ldots$, the tape contains $s(n)$ symbols from $\{1, \ldots, k\}$ in ascending order. To determine the number of such words of length $s(n)$, we consider the set of all words of length $s(n) + k - 1$ over the alphabet $\{A, |\}$ that contain exactly $k - 1$ symbols $|$. Each such word uniquely describes one possible content of the working tape: The symbols A represent the $s(n)$ symbols from $\{1, \ldots, k\}$, the symbols $|$ represent transitions from symbol i to symbol $i + 1$. Due to the sorting of the tape content, the assignment of the symbols A to the symbols from $\{1, \ldots, k\}$ is uniquely given by the position of the separators $|$. There are exactly

$$\binom{s(n)+k-1}{k-1}$$

words over $\{A, |\}$ of the described form and each of them corresponds to exactly one possible content of the working tape. Hence, M has at most

$$|Q| \cdot (n+2) \cdot (s(n)+1) \cdot \binom{s(n)+k-1}{k-1} \in \mathcal{O}(n \cdot (s(n))^k)$$

internal configurations in total.