Theory of Computer Science

G. Röger
Spring Term 2019

University of Basel
Computer Science

Exercise Sheet 8 — Solutions

Exercise 8.1 (Transitivity of Reductions, 1 point)

Show for any languages A, B and C: if $A \leq B$ and $B \leq C$, then $A \leq C$.

Solution:

Let $A \subseteq \Sigma_A^*$, $B \subseteq \Sigma_B^*$ and $C \subseteq \Sigma_C^*$. Since $A \leq B$, there is a total and computable function $f : \Sigma_A^* \to \Sigma_B^*$, with $x \in A$ iff $f(x) \in B$. From $B \leq C$ we analogously know that there is a function $g : \Sigma_B^* \to \Sigma_C^*$ with $x \in B$ iff $g(x) \in C$.
We define $h : \Sigma_A^* \to \Sigma_C^*$ as $g \circ f$ (i.e. $h(x) = g(f(x))$ for all $x \in \Sigma_A^*$). The function is total and computable, since the composition of total and computable functions is also total and computable. We now know that $x \in A$ iff $f(x) \in B$ iff $g(f(x)) \in C$ iff $h(x) \in C$. We conclude that A can be reduced (with h) to C: $A \leq C$.

Exercise 8.2 (Undecidability of the emptiness problem, 4 points)

The *emptiness problem* Emptiness for general (type-0) grammars is defined as:

Given a general grammar G, is $\mathcal{L}(G) = \emptyset$?

Prove that Emptiness is undecidable.

Solution:

We know from the hint: for every DTM M we can construct a grammar G_M with $\mathcal{L}(M) = \mathcal{L}(G_M)$, and for every grammar G we can construct a DTM M_G with $\mathcal{L}(G) = \mathcal{L}(M_G)$. Thus, the problems Emptiness and EmptinessDTM ("Is the language accepted by a given DTM empty?") can be reduced to one another: either both are decidable or both are undecidable. We now show that EmptinessDTM is undecidable.
Let Ω be the function that is always undefined, i.e. $\Omega(w)$ is undefined for all words w. According to Rice's theorem it is undecidable if a given DTM computes Ω. We call this problem ComputesUndef.
Let f be the function that handles a given word w as follows:

(a) Determine the DTM M_w that is encoded by w.

(b) Transform M_w into a modified DTM $\widetilde{M_w}$ that computes the same function as M_w but never stops with an invalid output, i.e. for every given input it either stops with a valid output or it does not stop at all.

(c) Return the encoding of $\widetilde{M_w}$.

This function f is a reduction of ComputesUndef to EmptinessDTM, because the DTM encoded by w computes the function Ω if and only if $\widetilde{M_w}$ accepts the empty language. Since ComputesUndef is undecidable, EmptinessDTM must be undecidable as well.

Exercise 8.3 (Undecidability of intersection problem, 1 point)

The *intersection problem* Intersection for general (type-0) grammars is defined as:

Given two general grammars G_1 and G_2, is $\mathcal{L}(G_1) \cap \mathcal{L}(G_2) = \emptyset$?

Show that Intersection is undecidable using a reduction and the fact that Emptiness is undecidable.

Solution:

We use a reduction from EMPTINESS to INTERSECTION. Let f be the function $f(G) = (G, G)$ for all G.

$$
\begin{aligned}
G \in \text{EMPTINESS} &\text{ iff } \mathcal{L}(G) = \emptyset \\
&\text{ iff } \mathcal{L}(G) \cap \mathcal{L}(G) = \emptyset \\
&\text{ iff } (G, G) \in \text{INTERSECTION} \\
&\text{ iff } f(G) \in \text{INTERSECTION}
\end{aligned}
$$

The function f is total and computable and reduces EMPTINESS to INTERSECTION. Since EMPTINESS is undecidable, INTERSECTION must be undecidable as well.