Lehrstuhl für Automatentheorie
Institut für Theoretische Informatik, TU Dresden
Prof. Dr. F. Baader

Nöthnitzer Str. 46
01187 Dresden
Tel.: 0351/463–38237
Tel.: 0351/463–39171

5. Exercises for the Course „Complexity and Logic“

Exercise 19:

Prove that the satisfiability of formulas of first-order predicate logic (with an equality predicate) is undecidable. To do this, use a reduction of the domino problem. How can you do it without equality?

Exercise 20:

Let L be a language. Then the *closure under Kleene Star of* L is defined as

$$L^* := \{w_1 \cdots w_k \mid k \geq 0 \wedge w_1, \ldots, w_k \in L\}.$$

Prove the following:

(a) If L, L' are in P, then $L \cup L'$ is in P.

(b) If L is in P, then L^* is in P.

Hint: For (b), it is convenient to use dynamic programming techniques.

Exercise 21:

For $A \subseteq \mathbb{N}$, define the languages $U(A) = \{1^n \mid n \in A\}$ and $B(A) = \{\widehat{n} \mid n \in A\}$, where $\widehat{n}$ denotes the binary encoding of n as in the lecture. Prove that $U(A) \in \mathsf{P}$ iff $B(A) \in \mathsf{DTime}(2^{O(n)})$.

Exercise 22:

A language L is *in* LogSpace if it is decided by a 2-tape Turing machine such that

- the first tape is read-only and contains the input and
- only $d \cdot \mathsf{log}(n)$ cells are used on the second tape on inputs of length n, for some $d \in \mathbb{N}$.

Prove that the following languages are in LogSpace:

(a) $L = \{a^n b^n \mid n \geq 0\}$;

(b) $L = \{w \in \{a, b\}^* \mid w$ is a palindrome, i.e., w read backwards is $w\}$.

Exercise 23:

Time and space are only two examples of complexity measures for computations. The following general approach is known as *Blum complexity*. Let f be a function mapping pairs (M, w), with M a (deterministic) Turing machine and w an input for M, to nonnegative integers. Then f is a *complexity measure* if it satisfies the following conditions:

(i) $f(M, w)$ is defined if and only if M terminates on w;

(ii) Given M, w, k, it is decidable whether $f(M, w) = k$.

Note that there may be M, w, k given as input such that M does not halt on w. Consider the following functions:

- $\mathit{space}'_M(w)$ is defined as the function $\mathit{space}_M(w)$ in the lecture, but it is undefined if M does not terminate on w.
- $\mathit{ink}_M(w)$ is the number of times during the finite computation of M on w that a symbol has to be overwritten by a different symbol. It is undefined if M does not terminate on w.
- $\mathit{carbon}_M(w)$ is the number of times during the finite computation of M on w that a symbol has to be overwritten by the same symbol. It is undefined if M does not terminate on w.

Show the following:

(a) $\mathit{time}_M(w)$ and $\mathit{space}'_M(w)$ are complexity measures.

(b) $\mathit{ink}_M(w)$ is a complexity measure.

Hint: the proof of (ii) involves show the following: if $\mathit{ink}_M(w) = k$, then the number of different configurations that the computation of M on w may take is finite.

(c) $\mathit{carbon}_M(w)$ is not a complexity measure.

Hint: To prove that (ii) is violated, reduce the word problem to the problem “$\mathit{carbon}(M, x) = k$?”.