University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
D. Atalay, S. Faour

Theoretical Computer Science - Bridging Course
Sample Solution Exercise Sheet 7

Due: Tuesday, 2nd of December 2025, 12:00 pm

Exercise 1: The Halting Problem Revisited *(3+3 Points)*

Show that both the halting problem and its special version are both undecidable.

(a) The *halting problem* is defined as

$$H = \{\langle M, w\rangle \mid \langle M\rangle \text{ encodes a TM and } M \text{ halts on string } w\}.$$

Hint: Assume H is decidable and try to reach a contradiction by showing that some known undecidable problem (cf. from the lecture) is decidable.

(b) The *special halting problem* is defined as

$$H_s = \{\langle M\rangle \mid \langle M\rangle \text{ encodes a TM and } M \text{ halts on } \langle M\rangle\}.$$

Hint: Assume that M is a TM which decides H_s and then construct a TM which halts iff M does not halt. Use this construction to find a contradiction.

Sample Solution

(a) The solution is via the reduction method and thus using the hint.
Assume H is decidable, hence there exists a TM D that decides on it.
We know from the lecture that the A_{TM} problem is undecidable.
We reach a contradiction by constructing a TM D' that decides on A_{TM} as follows.
D'= " On input $\langle M, w\rangle$, where M is a TM and w is a string:
1. Run TM D on $\langle M, w\rangle$, if D rejects, D' reject.
2. If D accepts, simulate M on w until it halts. If M accepts, accept; if M rejects, D' reject."
So D' is a decider for A_{TM}, but D' cannot exist.
Thus, our assumption is false; hence H is undecidable.

(b) The solution is via the diagonalization method and thus using the hint and similar to the lecture.
Assume H_s is decidable, hence there exists a TM D that decides on it.
We build a **Turing machine T, which is unlike the lecture not a decider**. We do so using the help of D to reach a contradiction. Note that the contradiction will stem from the idea that when feeding T its own encoding, we will reach a contradiction to some truth. However, our assumption yields that T should always be correct and never give us a contradiction, thus our assumption must be false and thus H is undecidable. We define T as follows.

T = "On input $\langle M\rangle$, where M is a TM:
1. Run D on $\langle M\rangle$
2. If D accepts, then T loop and if D rejects, then T halts and accepts.

Thus we have

$$D(\langle M\rangle) = \begin{cases} \text{accept} & \text{if } M \text{ halts } \langle M\rangle \\ \text{reject} & \text{if } M \text{ loops on } \langle M\rangle \end{cases}$$

And thus

$$T(\langle M\rangle) = \begin{cases} \text{loop} & \text{if } M \text{ halts } \langle M\rangle \\ \text{accept} & \text{if } M \text{ loops on } \langle M\rangle \end{cases}$$

We feed T its own encoding and reach a contradition as follows

$$T(\langle T\rangle) = \begin{cases} \text{loop} & \text{if } T \text{ halts } \langle T\rangle \\ \text{accept} & \text{if } T \text{ loops on } \langle T\rangle \end{cases}$$

Exercise 2: A Non-Turing Recongnizable Problem *(3 Points)*

Fix an enumeration of all Turing machines (that have input alphabet Σ): $\langle M_1\rangle, \langle M_2\rangle, \langle M_3\rangle, \dots$.
Fix also an enumeration of all words over Σ: $w_1, w_2, w_3, \dots$.
Prove that language $L = \{w \in \Sigma^* \mid w = w_i, \text{ for some } i, \text{ and } M_i \text{ does not accept } w_i\}$ is not Turing recognizable.
Hint: Try to find a contradiction to the existence of a Turing machine that recognizes L.

Sample Solution

Suppose that L is Turing recognizable.
Then there exists a Turing machine T that recognizes L. [1]
Then for the fixed enumerations of all turing machines, there must exist an index i such that $T = M_i$.
and thus for the fixed enumerations of all words, w_i will be the corresponding string for M_i.
Now for the specific string w_i, we have that T accepts w_i if and only if $w_i \in L$; that is,

$$T \text{ accepts } w_i \iff M_i \text{ does not accept } w_i.$$

But since $T = M_i$, this means that

$$M_i \text{ accepts } w_i \iff M_i \text{ does not accept } w_i.$$

which is a contradiction. Therefore such a TM can not exist. Hence, L is not Turing recongnizable.

Exercise 3: $\mathcal{O}$-Notation Formal Proofs *(1+2+2 Points)*

Roughly speaking, the set $\mathcal{O}(f)$ contains all functions that are not growing faster than the function f when additive or multiplicative constants are neglected. Formally:

$$g \in \mathcal{O}(f) \iff \exists c > 0, \exists M \in \mathbb{N}, \forall n \geq M : g(n) \leq c \cdot f(n)$$

For the following pairs of functions, state whether $f \in \mathcal{O}(g)$ or $g \in \mathcal{O}(f)$ or both. Proof your claims (you do not have to prove a negative result $\notin$, though).

(a) $f(n) = 100n,\ g(n) = 0.1 \cdot n^2$

(b) $f(n) = \sqrt[3]{n^2},\ g(n) = \sqrt{n}$

(c) $f(n) = \log_2(2^n \cdot n^3),\ g(n) = 3n$ *Hint: You may use that $\log_2 n \leq n$ for all $n \in \mathbb{N}$.*

[1] Which means that for any string $w \in \Sigma^*$, T accepts w if and only if $w \in L$; that is, for any string $w \in \Sigma^*$, there exists an index k such that $w := w_k$ and $\big(T \text{ accepts } w_k \iff M_k \text{ does not accept } w_k\big)$.

Sample Solution

(a) It is $100n \in \mathcal{O}(0.1n^2)$. To show that we require constants c, M such that $100n \leq c \cdot 0.1n^2$ for all $n \geq M$. Obviously this is the case for $c = 1000$ and $M = 1$.

(b) We have $g(n) \in O(f(n))$. Let $c := 1$ and $M := 1$. Then we have

$$\begin{aligned} & g(n) \leq c \cdot f(n) && (1) \\ \Leftrightarrow\quad & \sqrt{n} \leq n^{2/3} && (2) \\ \Leftrightarrow\quad & 1 \leq n^{1/6} && (3) \\ \Leftrightarrow\quad & 1 \leq n && (4) \end{aligned}$$

The last inequality is satisfied because $n \geq M = 1$.

(c) $f(n) \in O(g(n))$ holds. We give $c > 0$ and $M \in \mathbb{N}$ such that for all $n \geq M : \log_2(2^n \cdot n^3) \leq c \cdot n$. Indeed,

$$\begin{aligned} & \log_2(2^n \cdot n^3) \\ &= \log_2(2^n) + \log_2(n^3) \\ &= n + 3 \cdot \log_2(n) \\ &\leq n + 3n = 4n. \end{aligned}$$

Thus $\log_2(2^n \cdot n^3) \leq c \cdot 3n$ for $n \geq M := 1$ and $c := 4/3$.

We also have that $g(n) \in O(f(n))$ holds because

$$g(n) = 3n \leq 3(n + 3 \cdot \log_2(n)) = 3(\log_2(2^n \cdot n^3)) = 3 \cdot f(n).$$

Thus with $c = 3$ and for $n \geq M := 1$ we have $g(n) \leq cf(n)$.

Exercise 4: Sort Functions by Asymptotic Growth *(6 Points)*

Give a sequence of the following functions sorted by asymptotic growth, i.e., for consecutive functions g, f in your sequence, it should hold $g \in \mathcal{O}(f)$. Write "$g \cong f$" if $f \in \mathcal{O}(g)$ *and* $g \in \mathcal{O}(f)$.

$\log_2(n!)$	$\sqrt{n}$	2^n	$\log_2(n^2)$
3^n	n^{100}	$\log_2(\sqrt{n})$	$(\log_2 n)^2$
$\log_{10} n$	$10^{100} \cdot n$	$n!$	$n \log_2 n$
$n \cdot 2^n$	n^n	$\sqrt{\log_2 n}$	n^2

Sample Solution

For clarification, we write $g \lesssim f$ if $g \in \mathcal{O}(f)$, but not $f \in \mathcal{O}(g)$.

$$\begin{array}{lllllllll} & \sqrt{\log_2 n} & \lesssim & \log_2(\sqrt{n}) & \cong & \log_{10} n & \cong & \log_2(n^2) \\ \lesssim & (\log_2 n)^2 & \lesssim & \sqrt{n} & \lesssim & 10^{100} n & \lesssim & n \log_2 n \\ \cong & \log_2(n!) & \lesssim & n^2 & \lesssim & n^{100} & \lesssim & 2^n \\ \lesssim & n \cdot 2^n & \lesssim & 3^n & \lesssim & n! & \lesssim & n^n \end{array}$$