Gruppe	Hemmecke (10:15)	Hemmecke (11:00)	Popov
Name		Matrikel	SKZ

Klausur 2

Berechenbarkeit und Komplexität

12. Januar 2018

Part 1 RecFun2017

Let $f, g : \mathbb{N} \to_P \mathbb{N}$ be two partial functions that are defined as follows:

$$f(x) = \begin{cases} 0 & \text{if } x = 0, \\ \text{undefined} & \text{if } x = 1, \\ 2f(x-2) & \text{otherwise} \end{cases} \qquad g(x) = \begin{cases} x^2 & \text{if } x \text{ is odd,} \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Let $F(x) = f(f(2x))$ and $h(x) = f(x) + g(x)$.

1		no	*Is f primitive recursive?*
2	yes		*Is F primitive recursive?*

Obviously, is f defined for even input and has an even result in these cases. Thus $f(2x) = 2f(2(x-1)) = \cdots = 2^x f(0) = 2^x$. Therefore, $F(x) = f(2 \cdot 2^{x-1}) = 2^{2^{x-1}}$.

3	yes		*Is g μ-recursive?*
4	yes		*Is h μ-recursive?*

A nowhere defined function is, of course, μ-recursive.

5		no	*Can every total function of type $\mathbb{N} \to \mathbb{N}$ be computed by a Turing machine?*

Let $H : \mathbb{N} \to \mathbb{N}$ be the function that checks whether the input n is the code of a Turing machine. If it is not then $H(n) = 2$. If it is then $H(n)$ returns 1 if the TM corresponding to n halts on the empty input and returns 0, if that TM does not halt. Clearly, H is a total function, but if it were Turing computable, then the restricted Halting problem would be decidable.

Part 2 Grammar2017

Consider the grammar $G = (N, \Sigma, P, S)$ where $N = \{S, A\}$, $\Sigma = \{0, 1\}$, $P = \{S \to 0AA, 00A \to 10A, A \to 0A, A \to 0\}$.

6		no	*Is $L(G)$ finite?*

Consider the rule $A \to 0A$.

7	yes		*Is $1000 \in L(G)$?*

$S \to 0\underline{A}A \to \underline{00A}A \to 10\underline{A}A \to 100\underline{A} \to 1000$

8		no	*Is the grammar G context-free?*
9	yes		*Is there a Turing maching M such that $L(M) = L(G)$?*
10		no	*Does for every Turing machine M' exist a contest-sensitive grammar G' such that $L(M') = L(G')$?*

see Chomsky hierarchy

Part 3 Decidable2017

Consider the following problems. In each problem below, the input of the problem is the code $\langle M \rangle$ of a Turing machine $M = (Q, \Gamma, \sqcup, \{0, 1\}, \delta, q_0, F)$.

Problem A: *Does $L(M)$ contain the word 2017 in binary expansion?*
Problem B: *Does there exist a grammar G such that $L(M) = L(G)$.*
Problem C: *Is there a Turing machine M' with $L(M') \neq L(M)$*
Problem D: *Does there exist some word w such that M accepts w?*

11		no

Is A decidable?

Rice Theorem.

12	yes	

Is B semi-decidable?

A language generated by an unrestricted grammars is recursively enumerable and vice verse. So the question actually is whether the problem "true" is (semi-)decidable. That problem is even decidable, namely by a Turing machine that always return "yes" no matter what its input is.

13	yes	

Is C decidable?

There are infinitely many recursively enumerable languages. Among them is certainly a language $L \neq L(M)$. Since L is recursively enumerable, there exists a Turing machine M' with $L = L(M')$. So the answer to problem C is always "yes", and that is decidable.

14	yes	

Is D semi-decidable?

Run M (in parallel) on all words (usual trick of doing one step of the run of all instances of M and starting a new instance of M on the next word). Whenever an instance halts in an accepting state, the answer to problem D is "yes".

15		no

Let $P, P' \subseteq \{0,1\}^*$ and let M be a Turing machine that for every $w \in P$ computes a word $w' \in P'$ and for every $w \notin P$ computes a word $w' \notin P'$. Assume P is decidable. Can it in general be concluded that P' is decidable?

We have $P(w) \iff P'(f(w))$ where f is the "computable function" (that is required in Definition 42) computed by M. Thus $P \leq P'$. We would need the "other" direction, namely $P' \leq P$, i. e., a computable function g such that $P(g(w)) \iff P'(w)$ in order to apply Theorem 32 (lecture notes). Since nothing about such a g is known and its existance cannot be concluded from f, it cannot be concluded that P' is decidable.

Part 4 $\boxed{\textit{Complexity2017}}$
Let $f(n) = 20^n + 17^n$, $g(n) = (20 + 17)^n$, and $h(n) = n^{20} + n^{17}$.

16		no	*Is it true that $f(n) = \Theta(g(n))$?*
17	yes		*Is it true that $h(n) = O(f(n))$?*
18		no	*Is it true that $2^{h(n)} = O(g(n))$?*
19	yes		*Is it true that $\frac{1}{n} = O(\frac{1}{10^8})$?*

Part 5 $\boxed{\textit{LoopWhile2017}}$
Let P be a WHILE program that computes a total function $f : \mathbb{N} \to \mathbb{N}$ where x_1 is the input of the program P and x_0 its output. Let W be the following WHILE program that computes a function $g : \mathbb{N} \to \mathbb{N}$.

loop x_1 ***do*** P; $x_1 := x_0 + 1$ ***end***;

*Furthermore, let P' and W' be the programs that are obtained from P and W by replacing every **loop** by **while**, respectively. Let f' and g' the functions that are computed by P' and W' respectively.*

20	yes	

Is f Turing-computable?

Since f WHILE-computable.

21	yes	

Additionally assume that f is primitive recursive. Can it be concluded that g is LOOP-computable?

Since f is primitive recursive, there exists a LOOP program P_L that computes f. Then the program W_L (which is like W with P replaced by P_L) is a LOOP program that computes g.

22		no

If f' is a total function, can it be concluded that there exists a LOOP program that computes f'?

A **loop** statement can always be rewritten into an equivalent **while** statement. Let X be a program that computes $\mathrm{ack}(x, x)$. Let P be as X but rewritten to contain no **loop** statement. Then $P' = P$. Thus f' is total, but not primitive recursive.

23	yes	

Is the problem "$n \in \mathrm{range}(g')$" (i.e., "Does there exist some $m \in \mathbb{N}$ such that $g'(m) = n$?") decidable?
(Formally: Let $b : \mathbb{N} \to \{0, 1\}^*$ be the (Turing-computable) function that takes a natural number n as input and returns the binary representation of n. Is the set $R = \left\{ b(n) \in \{0, 1\}^* \;\middle|\; n \in \mathrm{range}(g') \right\}$ decidable?)

Obviously W' does not execute the body of the the outermost **while** if $x_1 = 0$. In that case $x_0 = 0$ is the result. In all other cases W' does not terminate. Therefore, $R = \{0\}$ and that set is finite and thus decidable.

Part 6 $\boxed{\textit{OpenComputability2017}}$

The syntax of a LOOP program is given by:

$$P ::= x_i = 0 \mid x_i := x_j + 1 \mid x_i := x_j - 1 \mid P; P \mid \textbf{loop } x_i \textbf{ do } P \textbf{ end}$$

Please note that the arithmetic operation allowed in a LOOP program are only $x_i := x_j + 1$ and $x_i := x_j - 1$.

24	*1 Point*

Write a LOOP program that computes the function $c(n) = \sum_{k=1}^{n} k$.

$x_0 := x_1 + 1;$
$x_0 := x_0 - 1;$ $\quad //x_0 := x_1$
loop x_1 do $\quad //\sum_{x_1 \ldots 0}$
$\quad x_1 := x_1 - 1;$
$\quad$ loop x_1 do $x_0 := x_0 + 1;$ end;
end;

25	*1 Point*

Determine an asymptotic lower bound $B(n)$ for the number of of executions of commands of the form $x_i := x_j + 1$ and $x_i := x_j - 1$ for any LOOP program that computes $c(n)$. Use Ω notation.
$B(n) = \Omega(\qquad\qquad)$

The result $c(n) = \frac{n(n+1)}{2}$ can only be achieved by executing at least $\Omega(n^2)$ times a command of the form $x_i := x_j + 1$.