Grundzüge der Theoretischen Informatik, WS 21/22: Musterlösung zum 7. Übungsblatt

Julian Dörfler

Aufgabe A7.1 (Indexmengen) *(4 Punkte + 2 Bonuspunkte)*
Welche der folgenden Sprachen sind Indexmengen? Beweisen Sie Ihre Antworten.

(a) $L_1 = \{i \in \mathbb{N} \mid \operatorname{im} \varphi_i \text{ ist endlich}\}$

(b) $L_2 = \{i \in \mathbb{N} \mid \varphi_i^0() = i\}$. Hierbei definieren wir $\varphi_i^0()$ als die Ausführung von i ohne Parameter.

(c) $L_3 = \{i \in \mathbb{N} \mid \operatorname{im} \varphi_i \text{ ist abzählbar}\}$

(d) $L_4 = \{i \in \mathbb{N} \mid \varphi_i(42) = 1337\}$

(e) $L_5 = \{i \in \mathbb{N} \mid \forall m \in \mathbb{N} : \varphi_i(m) \neq 42\}$

(f) $L_6 = \{i \in \mathbb{N} \mid i \text{ hält auf Eingabe 42 und } i \leq 10\}$

Lösung A7.1 (Indexmengen)
Für alle Sprachen von denen wir zeigen wollen, dass Sie Indexmengen sind, nehmen wir an, dass wir Gödelisierungen i und j gegeben haben, so dass i in der Sprache enthalten ist und $\varphi_i = \varphi_j$.

(a) L_1 ist eine Indexmenge: Wenn $\varphi_i = \varphi_j$ ist $\operatorname{im} \varphi_j$ ebenfalls endlich und daher $j \in L_1$.

(b) L_2 ist keine Indexmenge. Aus Aufgabe A7.6(a) folgt, dass L_2 nicht leer ist. Sei also $i \in L_2$. Sei ferner P ein Programm, welches erst $x_i := x_i$ (oder irgendeinen anderen terminierenden Unsinn) ausführt und dann i simuliert. Dann gilt, dass $j := \text{göd}(P) \neq i$, aber $\varphi_j = \varphi_i$ und daher $\varphi_j^0() = i \neq j$. j ist also nicht in L_2 enthalten und daher ist L_2 keine Indexmenge.

(c) L_3 ist eine (triviale) Indexmenge, da $\operatorname{im} \varphi_i \subseteq \mathbb{N}$ *immer* abzählbar ist.

(d) L_4 ist eine Indexmenge: Wenn $\varphi_i = \varphi_j$ ist $\varphi_j(42) = \varphi_i(42) = 1337$ und daher $j \in L_4$.

(e) L_5 ist eine Indexmenge: Wenn $\varphi_i = \varphi_j$, dann gilt für alle $m \in \mathbb{N}$ direkt $\varphi_j(m) = \varphi_i(m) \neq 42$ und daher ist j ebenfalls in L_5.

(f) Diese Aufgabe hängt sehr stark von der verwendeten Gödelisierung ab. Da L_6 endlich ist, gilt dass $L_6 \in \mathsf{REC}$. Wäre L_6 eine Indexmenge, müsste sie daher - nach dem Satz von Rice - trivial sein.

Schon das Programm mit Gödelnummer 0 ($x_0 := x_0 + x_0$) terminiert aber auf jeder Eingabe, also auch auf 42, somit ist L_6 nicht leer.

Somit kann L_6 keine Indexmenge sein.

Aufgabe A7.2 (Abschlusseigenschaften) *(2 Punkte)*
Eine Menge M von Teilmengen von $\mathbb{N}$ ist abgeschlossen unter

- Komplement, wenn für alle $A \in M$ gilt, dass $\bar{A}$ in M ist.
- Schnitt, wenn für alle $A, B \in M$ gilt, dass $A \cap B$ in M ist.
- Vereinigung, wenn für alle $A, B \in M$ gilt, dass $A \cup B$ in M ist.

(a) Untersuchen Sie, inwieweit REC unter Komplement, Schnitt und Vereinigung abgeschlossen ist.

(b) Untersuchen Sie, inwieweit RE unter Komplement, Schnitt und Vereinigung abgeschlossen ist.

Lösung A7.2 (Abschlusseigenschaften)

(a) REC ist unter Komplement, Schnitt und Vereinigung abgeschlossen. Da χ_A und χ_B total sind, führen wir das `WHILE`-Programm für χ_A und χ_B nacheinder aus und erhalten jeweils 0 oder 1. Mit diesen Werten können wir dann die entsprechenden Funktionen direkt berechnen.

(b) RE ist abgeschlossen unter Schnitt und Vereinigung. Schnitt und Vereinigung können erreicht werden indem die zwei Programme interleaved ausgeführt werden (für jeden Schritt den das eine Programm macht, macht das andere auch einen bevor wir fortfahren mit der Simulation). RE ist nicht abgeschlossen unter Komplement, Gegenbeispiel ist $\overline{H_0} \notin \mathsf{RE}$, aber $H_0 \in \mathsf{RE}$.

Aufgabe A7.3 (Rekursionstheorem) *(2 Punkte)*
Zeigen Sie, dass die Menge $A = \{i \in \mathbb{N} \mid \forall x \in \mathbb{N} : \ \varphi_i(x) = i + 1\}$ nicht leer ist.

Lösung A7.3 (Rekursionstheorem) Die Funktion $f(g, m) = g + 1$ ist offensichtlich `WHILE`-berechenbar. Damit gibt uns das Rekursionstheorem ein $e \in \mathbb{N}$ mit $\varphi_e(m) = f(e, m) = e + 1$, also $e \in A$. Somit ist A nicht leer.

Aufgabe A7.4 (Reduktionen) *(4 Punkte)*
Wir definieren

$$L = \{i \in \mathbb{N} \mid \operatorname{dom} \varphi_i \text{ ist endlich}\}$$

(a) Zeigen Sie: $L \notin \mathsf{RE}$

(b) Zeigen Sie: $L \notin \mathsf{co\text{-}RE}$

Lösung A7.4 (Reduktionen)

(a) Wir reduzieren von $\overline{H_0}$ auf L. Gegeben g, geben wir die Gödelisierung des folgenden Programms P_g aus:

Gegeben m, simuliere g auf g. Danach gib m aus.

Sei nun $g \in \overline{H_0}$. Dann hält g auf Eingabe g nicht. In diesem Fall hält P_g bei keiner Eingabe, also $\operatorname{dom} \varphi_{P_g} = \emptyset$, also $p \in L$.
Sei nun $g \notin \overline{H_0}$. Dann hält g bei Eingabe g. Daher hält P_g auf allen Eingaben und es gilt $\operatorname{dom} \varphi_{P_g} = \mathbb{N}$, also $p \notin L$.

Da $\overline{H_0} \notin \mathsf{RE}$ ist und unsere Reduktion offensichtlich `WHILE`-berechenbar ist, ist also auch $L \notin \mathsf{RE}$.

(b) Wir reduzieren von H_0 auf L. Gegeben g, geben wir die Gödelisierung des folgenden Programms P_g aus:

Gegeben m, simuliere g auf g für m Schritte. Falls diese Simulation innerhalb der m Schritte terminiert divergiere, ansonsten gib m aus.

Sei nun $g \in H_0$. Dann hält g auf Eingabe g nach $t \in \mathbb{N}$ Schritten. Daher hält P_g bei Eingabe $m \geq t$ nicht und berechnet für $m < t$ die Identitätsfunktion. Es gilt also $\operatorname{dom} \varphi_{P_g} = \{0, 1, \ldots, t-1\}$, also $\text{göd}(P_g) \in L$.
Sei nun $g \notin H_0$. Dann hält g bei Eingabe g niemals. Somit berechnet P_g für alle $m \in \mathbb{N}$ die Identitätsfunktion und es gilt $\operatorname{dom} \varphi_{P_g} = \mathbb{N}$, also $\text{göd}(P_g) \notin L$.

Da $H_0 \notin \mathsf{co\text{-}RE}$ ist und unsere Reduktion offensichtlich `WHILE`-berechenbar ist, ist also auch $L \notin \mathsf{co\text{-}RE}$.

Alternativer Beweis für beide Teilaufgaben auf einmal: Wir reduzieren von $\overline{V_0}$ auf L. Gegeben g, geben wir die Gödelisierung des folgenden Programms P_g aus:

Gegeben m, simuliere g nacheinander auf allen Eingaben $m' \leq m$. Falls alle diese Simulationen mit Ausgabe 0 terminieren gebe m aus, ansonsten divergiere.

Sei nun $g \in \overline{V_0}$. Dann gibt es eine kleinste Eingabe n, so dass $\varphi_g(n)$ undefiniert ist, oder $\varphi_g(n) \neq 0$. Daher hält P_g bei Eingabe $m \geq n$ nicht und berechnet für $m < n$ die Identitätsfunktion. Es gilt also $\operatorname{dom} \varphi_{P_g} = \{0, 1, \ldots, n-1\}$, also $\text{göd}(P_g) \in L$.
Sei nun $g \notin \overline{V_0}$. Dann hält g bei jeder Eingabe mit Ausgabe 0. Somit berechnet P_g für alle $m \in \mathbb{N}$ die Identitätsfunktion und es gilt $\operatorname{dom} \varphi_{P_g} = \mathbb{N}$, also $\text{göd}(P_g) \notin L$.
Da $\overline{V_0}$ weder in RE, noch in $\mathsf{co\text{-}RE}$ ist und die Reduktionsfunktion `WHILE`-berechenbar ist, kann also auch L weder in RE, noch in $\mathsf{co\text{-}RE}$ sein.

Aufgabe A7.5 (**REC, RE und co-RE**) *(4 Punkte)*
Wir definieren co-RE als die Menge aller Sprachen, deren Komplement rekursiv aufzählbar ist, in anderen Worten:

$$L \in \mathsf{co\text{-}RE} :\Leftrightarrow \overline{L} \in \mathsf{RE}\,.$$

Entscheiden Sie ob die Sprachen L_3, L_4, L_5 und L_6 aus Aufgabe A7.1 in REC, RE und/oder in $\mathsf{co\text{-}RE}$ enthalten sind und beweisen Sie Ihre Antworten.
Tipp: Nutzen Sie (unter anderem) den Satz von Rice.

Lösung A7.5 (REC, RE **und** co-RE)

- L_3 ist eine triviale Indexmenge, und damit in REC enthalten. Daraus folgt, dass L_3 auch in RE und $\mathsf{co\text{-}RE}$ enthalten ist.

- L_4 ist eine Indexmenge, die nicht trivial ist, da konstante Funktionen in L_4 sind, aber z.B. das Programm, das nie terminiert nicht. Also ist $L_4 \notin \mathsf{REC}$ nach dem Satz von Rice. L_4 ist rekursiv aufzählbar, da ein WHILE-Programm P bei Eingabe i, i auf Eingabe 42 simulieren kann. Wenn die Simulation mit Ausgabe 1337 hält, geben wir 1 aus, ansonsten divergieren wir. P berechnet dann χ'_{L_4}, also ist $L_4 \in \mathsf{RE}$. Daher muss gelten, dass $L_4 \notin \mathsf{co\text{-}RE}$, da ansonsten $L_4 \in \mathsf{REC}$.

- L_5 ist eine Indexmenge, die nicht trivial ist, da das Programm, dass nie terminiert nicht in L_5 ist, aber die Konstante 42-Funktion schon. Also ist $L_5 \notin \mathsf{REC}$ nach dem Satz von Rice. $\overline{L}_5$ ist rekursiv aufzählbar. Der Übersicht halber geben wir $\overline{L}_5$ explizit an:
$$\overline{L}_5 = \{i \in \mathbb{N} \mid \exists m \in \mathbb{N} : \varphi_i(m) = 42\}$$
Folgendes `WHILE`-Programm P berechnet die Funktion$\chi'_{\overline{L}_5}$:
Wir testen für alle Paare (m, t) (dies machen wir, indem wir über alle natürlichen Zahlen n iterieren und dann $m = \pi_1(n)$ und $t = \pi_2(n)$ setzen), ob $\varphi_i(m)$ nach $\leq t$ Schritten 42 ausgibt. Falls wir ein solches Paar finden, geben wir 1 aus.

  Ist nun $i \in \overline{L}_5$, so gibt es eine Eingabe m und eine Schrittzahl t, so dass i nach t Schritten mit Ausgabe 42 auf m terminiert, somit gibt P nach Betrachtung von (m, t) dann 1 aus. Ist $i \notin \overline{L}_5$, so gibt es keine Eingabe, so dass i mit Ausgabe 42 terminieren würde, P wird also nicht terminieren.

  $\overline{L}_5$ ist also in RE, daher gilt dass $L_5 \in \mathsf{co\text{-}RE}$ und ferner auch $L_5 \notin \mathsf{RE}$, da sonst $L_5 \in \mathsf{REC}$.

- L_6 ist eine endliche Menge und damit in REC enthalten. Daraus folgt, dass L_6 auch in RE und $\mathsf{co\text{-}RE}$ enthalten ist.

Aufgabe A7.6 (Quines und Kolmogorow-Komplexität) *(4 Bonuspunkte)*

(a) Die Sprache L_2 aus Aufgabe A7.1 ist die Menge aller *Quines*, d.h. Programme die ihren eigenen Quellcode[1] ausgeben. Zeigen Sie, dass es unendlich viele Quines gibt, also für jedes $c \in \mathbb{N}$ eine Quine $g \geq c$ existiert, so dass $g \in L_2$.

[1]In unserem Modell ist die Gödelisierung eines WHILE-Programms dessen Quellcode.

(b) In dieser Aufgabe sind wir an der *Komplexität* einer Zahl x interessiert. Intuitiv ist diese die Länge der kürzesten Beschreibung der Zahl x. Sei beispielsweise

$x =$ 373 391 848 741 020 043 532 959 754 184 866 588 225 409 776 783 734 007 750 636 931 722 079 040 617 265 251 229 993 688 938 803 977 220 468 765 065 431 475 158 108 727 054 592 160 858 581 351 336 982 809 187 314 191 748 594 262 580 938 807 019 951 956 404 285 571 818 041 046 681 288 797 402 925 517 668 012 340 617 298 396 574 731 619 152 386 723 046 235 125 934 896 058 590 588 284 654 793 540 505 936 202 376 547 807 442 730 582 144 527 058 988 756 251 452 817 793 413 352 141 920 744 623 027 518 729 185 432 862 375 737 063 985 485 319 476 416 926 263 819 972 887 006 907 013 899 256 524 297 198 527 698 749 274 196 276 811 060 702 333 710 356 481.

Nun können wir, wie oben, x einfach aufschreiben. Diese Beschreibung ist jedoch, wie wir sehen, recht lange. Tatsächlich gibt es eine kürzere Beschreibung:

$$x = 3^{4^5}.$$

Nun soll gezeigt werden, dass die Komplexität einer Zahl nicht berechenbar ist. Dazu definierte der sowjetische Mathematiker Andrei Kolmogorow 1965 die *Kolmogorow-Komplexität* einer Zahl anhand der Länge des kürzesten Programmes, welches die Zahl ausgibt. In unserem Modell benutzen wir die Gödelisierungsfunktion:

$$K(x) := \min\{i \in \mathbb{N} \mid \varphi_i^0() = x\}.$$

Ihre Aufgabe ist es nun, zu zeigen, dass $K \notin \mathsf{R}$ [2]. Nehmen Sie dazu an, dass $K \in \mathsf{R}$ - dann existiert ein `WHILE`-Programm `K`, welches K berechnet - und nutzen Sie dann das folgende WHILE-Programm `P` um die Annahme zu einem Widerspruch zu führen.

Input: t
  $x := 0$
  while $\mathtt{K}(x) \leq t$ **do**
    $x++$
  od
Output: x

Hinweis: Nutzen Sie für beide Aufgabenteile das Rekursions-Theorem.

Lösung A7.6 (Quines und Kolmogorow-Komplexität)

(a) Sei $\mathsf{id} : \mathbb{N}^1 \to \mathbb{N}$ die Identitätsfunktion. Wir können id durch `WHILE`-Programme mit beliebig hoher Gödelnummer g berechnen (zum Beispiel durch die c-fache Konkatenation von $x_1 := x_1 + x_1$). Nun können wir das Rekursions-Theorem anwenden und erhalten ein $g' \in \mathbb{N}$, so dass

$$\varphi_{g'}^0() = \mathsf{id}(g') = g'.$$

[2] R ist die Menge aller `WHILE`-berechenbaren Funktionen (vgl. Skript Definiton 7.6). Machen Sie sich klar, dass $\mathsf{R} \neq \mathsf{REC}$.

Folglich ist $g' \in L_2$ und damit $L_2 \neq \emptyset$. Es stellt sich heraus, dass das Rekursionstheorem die Gödelnummern nur vergrößert, also ebenso $g' \geq g$ gilt.

Alternativer Beweis: Wir definieren uns

$$p = 1 + \max_{i \leq c} \varphi_i^0().$$

p ist somit also eine Zahl, die nicht von Programmen mit Gödelnummer $\leq c$ ohne Eingabe berechnet werden kann. Da wir nur die Ausgabe von endlich vielen Programmen betrachten, ist $p \in \mathbb{N}$ wohldefiniert.

Wir betrachen nun

$$f(x) := \begin{cases} x & \text{wenn } x > c \\ p & \text{sonst} \end{cases}$$

Diese Funktion ist `WHILE`-berechenbar, da wir p einfach hardcoden können.

Nun gibt uns das Rekursionstheorem die Existenz einer Gödelisierung g, sodass $\varphi_g^0() = f(g)$. Wir beweisen nun, dass $g > c$ ist. Nehmen wir dafür an $g \leq c$ gilt. Dann ist $\varphi_g^0() = p \geq 1 + \varphi_g^0()$, was ein Widerspruch ist. Somit gilt: $g > c$ und somit $\varphi_g^0() = f(g) = g$.

Alternativer Beweis 2: Wir definieren uns für alle $i \in \mathbb{N}$ paarweise verschiedene Funktionen

$$f_i(x, y) := \begin{cases} x & \text{wenn } y = 0 \\ i & \text{sonst} \end{cases}$$

Diese Funktion ist `WHILE`-berechenbar, also gibt uns das Rekursionstheorem die Existenz einer Gödelisierung g_i, sodass $\varphi_{g_i}^1(y) = f_i(g, y)$. Nun gilt: $\varphi_{g_i}^0() = \varphi_{g_i}^1(0) = f_i(g_i, 0) = g_i$, womit g_i eine Quine ist. Nun berechnen aber alle g_i unterschiedliche Funktionen (z.B auf Eingabe 1), daher gibt es unendlich viele Quines.

(b) Wir führen die Annahme, dass `K` aus der Aufgabenstellung existiert, zu einem Widerspruch. Sei $f = \varphi_{\mathtt{P}}$. Dann ist f WHILE-berechenbar. Bevor wir das Rekursions-Theorem anwenden untersuchen wir `P`: Wenn das Programm bei Eingabe t eine Zahl x ausgibt, dann gilt, dass x die kleinste Zahl mit der Eigenschaft $K(x) > t$ ist. Wir wollen nun zeigen, dass `P` immer eine Ausgabe hat bzw. dass f total ist, was äquivalent dazu ist, dass es für jede Zahl t ein x mit $K(x) > t$ gibt. Angenommen, dies sei nicht der Fall, existiert ein $T \in \mathbb{N}$ so dass für alle $x \in \mathbb{N}$ gilt, dass $K(x) \leq T$. Daraus folgt aber nun, dass es für alle natürlichen Zahlen x eine Gödelisierung i_x gibt, so dass $\varphi_{i_x}^0() = x$ und $i_x \leq T$. Dies ist ein Widerspruch, da es nur endlich viele Gödelisierungen $i_x \leq T$, aber unendlich viele $x \in \mathbb{N}$ gibt.
Nun wenden wir das Rekursions-Theorem auf f an und erhalten ein $g \in \mathbb{N}$, so dass gilt

$$\varphi_g^0() = f(g) = \min\{x \mid K(x) > g\}.$$

Da f total ist, existiert ein k, so dass $f(g) = k$. Nun gilt

(I) $K(k) \leq g$, da $\varphi_g^0() = f(g) = k$ und

(II) $K(k) > g$, da $k = f(g) = \min\{x \mid K(x) > g\}$

was ein Widerspruch ist. Die Annahme, dass `K` existiert bzw. dass K WHILE-berechenbar ist, war also falsch.