SS 2015

Zentralübung zur Vorlesung Theoretische Informatik

Dr. Werner Meixner

Fakultät für Informatik
TU München

http://www14.in.tum.de/lehre/2015SS/theo/uebung/

2. Juli 2015

ZÜ IX

Übersicht:

1. Übungsbetrieb	Hinweise, Fragen, Probleme?
2. Thema	Reduzierbarkeit von Problemen Universelle Turingmaschine Reduktion von Problemen
3. Vorbereitung	Blatt 11

1. Hinweis, Fragen

Terminänderung:

Die Zentralübung am 9. Juli findet statt von 15.15 bis 16.45 Uhr.

Aktuelle Fragen?

2. Thema Reduzierbarkeit von Problemen

2.1 Universelle Turingmaschine

Die Ausführung der Berechnungen einer beliebigen vorgelegten, binär kodierten Turingmaschine für eine beliebige vorgelegte binär kodierte Eingabe kann grundsätzlich mit Bleistift und Papier geschehen. Es gibt einen Algorithmus für diese Ausführung.

Da es zu jedem Algorithmus grundsätzlich eine Turingmaschine gibt, die dieses Programm ausführt, so gibt es eine sogenannte „Universelle Turingmaschine“ U, die jede Turingmaschine mit beliebiger Eingabe simulieren kann.

Diese Tatsache nennt man den
Satz von der Existenz einer universellen Turingmaschine.

Wir setzen die Gültigkeit dieses Satzes von nun an voraus.

2.2 Reduktion von Problemen

Eine Menge $A \subseteq \Sigma^*$ heißt reduzierbar auf eine Menge $B \subseteq \Gamma^*$ genau dann, wenn es eine

1. totale,
2. berechenbare,
3. Funktion $f : \Sigma^* \to \Gamma^*$ gibt, so dass gilt:
4. $\forall w \in \Sigma^* : \ w \in A \Leftrightarrow f(w) \in B$.

Intuitiv: Ein Problem in der Sprache A kann man durch Übersetzung in ein Problem in der Sprache B lösen, falls obige Bedingungen gelten.
Typisches Problem: Wortproblem

3. Vorbereitung Blatt 11

3.1 VA 1

1. Im Folgenden bezeichne $a(n, m)$ die Ackermann-Funktion. Berechnen Sie $a(1, 6)$ und $a(2, 1)$!
2. Sei a die Ackermann-Funktion. Zeigen Sie, dass die folgenden Wertemengen entscheidbar sind.

   (i) $W_a = \{a(n, m)\,;\, n, m \in \mathbb{N}_0\}\,.$

   (ii) $W'_a = \{a(n, n)\,;\, n \in \mathbb{N}_0\}\,.$

1. Im Folgenden bezeichne $a(n, m)$ die Ackermann-Funktion. Berechnen Sie $a(1, 6)$ und $a(2, 1)$!

Lösung

Die Rekursionsgleichungen liefern einerseits

$$\begin{aligned} a(1,6) &= a(0, a(1,5)) = a(1,5) + 1 \\ &= a(1,4) + 2 \\ &= a(1,3) + 3 \\ &= a(1,2) + 4 \\ &= a(1,1) + 5 \\ &= a(1,0) + 6 \\ &= a(0,1) + 6 = 2 + 6 = 8\,. \end{aligned}$$

Andererseits rechnen wir in ausführlicher Notation

$$
\begin{aligned}
a(2,1) &= a(1,a(2,0)) \\
&= a(1,a(1,1)) \\
&= a(1,a(0,a(1,0))) \\
&= a(1,a(0,a(0,1))) \\
&= a(1,a(0,2)) \\
&= a(1,3)) \\
&= a(0,a(1,2)) \\
&= a(0,a(0,a(1,1))) \\
&= a(0,a(0,a(0,a(1,0)))) \\
&= a(0,a(0,a(0,a(0,1)))) \\
&= a(0,a(0,a(0,2))) \\
&= a(0,a(0,3)) \\
&= a(0,4) \;=\; 5\,.
\end{aligned}
$$

2. Sei a die Ackermann-Funktion. Zeigen Sie, dass die folgenden Wertemengen entscheidbar sind.

$$\text{(i)} \quad W_a = \{a(n,m)\,;\, n,m \in \mathbb{N}_0\}\,.$$

$$\text{(ii)} \quad W'_a = \{a(n,n)\,;\, n \in \mathbb{N}_0\}\,.$$

Lösung

(i)

Es gilt $W_a \subseteq \mathbb{N}_0$. Grundsätzlich kann man Entscheidbarkeitsfragen auf die Definition zurückführen. Danach ist W_a genau dann entscheidbar, wenn die charakteristische Funktion $\chi_{W_a} : \mathbb{N}_0 \to \{0,1\}$ berechenbar ist.

Bekanntlich ist a eine berechenbare Funktion. Leider (oder nicht leider) ist es so, dass nicht jede berechenbare Funktion einen entscheidbaren Wertebereich besitzt. Wir müssen also den konkreten Wertebereich von a analysieren bzw. zeigen, dass für χ_{W_a} ein Algorithmus existiert.

Nun liefert uns bereits die erste definierende Gleichung der Ackermann-Funktion $a(0, n) = n + 1$ für alle $n \in \mathbb{N}_0$ den Beweis, dass $\mathbb{N} \subseteq W_a$ gilt. Tatsächlich gilt sogar $\mathbb{N} = W_a$, was aber nicht bewiesen werden muss, denn sicher gilt nun entweder $W_a = \mathbb{N}$ oder $W_a = \mathbb{N}_0$. In beiden Fällen ist χ_{W_a} total und berechenbar.

(ii)

Wir berechnen $\chi_{W'_a}(x) = c$ mit dem folgenden WHILE-Programm, wobei wir uns auf ein WHILE-Programm zur Berechnung der Ackermann-Funktion $a(n, n)$ stützen.

$x_0 := x$; $c_0 := 0$;
$n_0 := x_0$;
LOOP n_0 DO
$n_0 := n_0 - 1$;
$a_0 := a(n_0, n_0)$;
IF $a_0 = x_0$ THEN $c_0 := 1$ END END ;
$c := c_0$

Die Korrektheit des Programms ergibt sich aus der strengen Monotonie von $a(n, n)$ zusammen mit $n < a(n, n)$.

Mit den Eigenschaften der Ackermann-Funktion nach Vorlesung folgt die strenge Monotonie von a', insbesondere

$$a'(n) = a(n,n) < a(n,n+1) < a(n+1,n) < a(n+1,n+1) = a'(n+1)\,.$$

Nach Vorlesung gilt auch

$$n < a(n,n)\,.$$

3.2 VA 2

Sei $a : \mathbb{N}_0 \times \mathbb{N}_0 \to \mathbb{N}_0$ die Ackermann-Funktion.

1. Zeigen Sie, dass $f(m,n) := (a(m,n))^2$ nicht primitiv-rekursiv ist.
2. Die Funktion $a' : \mathbb{N}_0 \to \mathbb{N}_0$ sei gegeben durch $a'(n) = a(n,n)$.
   Sei $W_{a'} = \{a'(n)\,;\, n \in \mathbb{N}_0\}$.
   Zeigen Sie, dass die Umkehrfunktion $b' : W_{a'} \to \mathbb{N}_0$ von a' μ-rekursiv ist.

1. Zeigen Sie, dass $f(m,n) := (a(m,n))^2$ nicht primitiv-rekursiv ist.

Lösung

Wir nehmen an, f sei primitiv-rekursiv, und geben eine primitiv-rekursive Definition von a an. Dazu benötigen wir die (ganzzahlige) Quadratwurzelfunktion, die man leicht mit Hilfe der beschränkten Maximierung als primitiv rekursiv nachweisen kann:

$$sqrt(n) = max\ \{i \leq n\,;\ i^2 \dot{-} n = 0\}$$

Dann ist $a(m,n) = sqrt(f(m,n))$, und somit wäre a primitiv-rekursiv. . . ein Widerspruch.

2. Die Funktion $a' : \mathbb{N}_0 \to \mathbb{N}_0$ sei gegeben durch $a'(n) = a(n, n)$.
   Sei $W_{a'} = \{a'(n) \,;\, n \in \mathbb{N}_0\}$.
   Zeigen Sie, dass die Umkehrfunktion $b' : W_{a'} \to \mathbb{N}_0$ von a' μ-rekursiv ist.

Lösung

Die Ackermann-Funktion ist nach Vorlesung total und berechenbar, und folglich auch μ-rekursiv. Wegen $a'(n) = a(\pi_1^1(n), \pi_1^1(n))$ ist a' ebenfalls μ-rekursiv.
Nach Aufgabenstellung können wir von der Existenz einer eindeutigen Umkehrfunktion $b' : W_{a'} \to \mathbb{N}_0$ ausgehen. Die Existenz vo b' folgt z. B. aus der strengen Monotonie der Ackermann-Funktion.

Wir verwenden den μ-Operator, um die Funktion b' zu definieren.
Sei $h : \mathbb{N}_0 \times \mathbb{N}_0 \to \mathbb{N}_0$, mit

$$h(n, m) = (m \dot{-} a'(n)) + (a'(n) \dot{-} m) \ .$$

Die Definition ist gerade so gewählt, dass
$h(n, m) = 0 \iff a'(n) = m$ gilt.

Es gilt

$$(\mu h)(m)$$

$$= \begin{cases} \min\{n \in \mathbb{N}_0 \,;\, h(n,m) = 0\} & \text{falls ein solches } n \text{ existiert und } \underbrace{h(k,m) \neq \perp \text{ für alle } k \leq n \text{ gilt}}_{\text{immer erfüllt, da } h \text{ total ist}} \\ \perp & \text{sonst} \end{cases}$$

$$= \begin{cases} \min\{n \in \mathbb{N}_0 \,;\, a'(n) = m\} & \text{falls ein solches } n \text{ existiert} \\ \perp & \text{sonst} \end{cases}$$

$$= \begin{cases} b'(m) & \text{falls } m \in W_{a'} \\ \perp & \text{sonst} \end{cases}$$

3.3 VA 3

Wir betrachten die in der Vorlesung beschriebene Kodierung von Turingmaschinen durch Wörter über $\Sigma^* = \{0,1\}^*$.

Für ein $w \in \Sigma^*$ beschreibt $\varphi_w : \Sigma^* \to \Sigma^*$ dann die Funktion, die durch die Turingmaschine M_w berechnet wird.

Finden Sie informelle Beschreibungen für die folgenden Mengen:

1. $A = \{w \in \Sigma^* \,;\, \varphi_w = \Omega\}\,.$

2. $B = \{w \in \Sigma^* \,;\, \varphi_w(101) \neq \perp\}\,.$

Bemerkung: Die Standard-Turingmaschine, die für ein ,,ungeeignetes“ Kodewort w gewählt wird, terminiert nie und berechnet deshalb die nirgends definierte Funktion Ω.

(1) $A = \{w \in \Sigma^* \,;\, \varphi_w = \Omega\}$

Lösung

A ist die Menge aller (Kodes der) Turingmaschinen, die die überall undefinierte Funktion Ω berechnen, die also auf keiner Eingabe halten.

In A sind also auch alle w enthalten, die die Standard-Turingmaschine bezeichnen oder die keine Kodierung einer Turingmaschine darstellen.

(2) $B = \{w \in \Sigma^* \,;\, \varphi_w(101) \neq \perp\}$

Lösung

Die Menge aller (Codes der) Turingmaschinen, die auf der Eingabe 101 anhalten.

Bemerkung: Man beachte, dass die Mengen A und B wohldefiniert sind, unabhängig von der Frage ihrer Entscheidbarkeit.

3.4 VA 4

Zeigen Sie die Unentscheidbarkeit der folgenden Mengen und wenden Sie zum Beweis Techniken der Reduzierbarkeit eines Problems A auf ein Problem B an.

1. $H_{\Sigma^*} = \{w\,;\, M_w \text{ hält für mindestens eine Eingabe}\}\,.$

2. $C =$
   $\{w\,;\, M_w$ berechnet die Funktion g mit $g(n) = 0$ für alle n$\}\,.$

Bei der Lösung gehen wir davon aus (siehe Thema), dass es eine universelle Turingmaschine U gibt, die die Berechnungen jeder Turingmaschine M_w auf deren Eingabe x simulieren kann, und deshalb insbesondere genau dann hält, wenn M_w hält.

Außerdem wurde in der Vorlesung bewiesen, dass das Halteproblem auf leerem Band H_0 nicht entscheidbar ist.

(1) $H_{\Sigma^*} = \{w\,;\, M_w \text{ hält für mindestens eine Eingabe}\}$

Lösung

Wir reduzieren das bekannte Halteproblem H_0 auf das Problem H_{Σ^*} durch Konstruktion einer totalen und berechenbaren Funktion f wie folgt.

Es sei $w' = f(w)$ der Code einer Turingmaschine $M_{w'}$, die bei Eingabe eines Wortes y Folgendes ausführt:

Die Turingmaschine M_w wird bei leerer Eingabe simuliert (beispielsweise auf einem zweiten Band). Offenbar ist f berechenbar.

Es gilt: Falls M_w hält, dann hält auch $M_{w'}$.

Offenbar folgt nun $w \in H_0 \Leftrightarrow f(w) \in H_{\Sigma^*}$,
d.h. f reduziert H_0 auf H_{Σ^*}.

(2)
$C = \{w\,;\, M_w$ berechnet die Funktion g mit $g(n) = 0$ für alle n$\}$

Lösung

Wir verfahren analog zur Lösung der vorhergehenden Aufgabe und reduzieren mit Hilfe einer Funktion f das Problem H_0 auf C:

Es sei $w' = f(w)$ der Code einer Turingmaschine $M_{w'}$, die bei Eingabe eines Wortes y folgendes ausführt:

Zunächst wird die Turingmaschine M_w bei leerer Eingabe simuliert. Falls M_w hält, dann schreibt $M_{w'}$ eine 0 auf das Band und terminiert.

Offenbar gilt nun $w \in H_0 \Leftrightarrow f(w) \in C$,
d.h. f reduziert H_0 auf C.

Es ist klar, dass f berechenbar ist.