Technische Universität München
Fakultät für Informatik
Prof. Tobias Nipkow, Ph.D.
Dr. Werner Meixner, Dr. Alexander Krauss

Sommersemester 2010
Lösungsblatt 10
9. Juli 2010

Einführung in die Theoretische Informatik

Hausaufgabe 1 (5 Punkte)

Die konstante Funktion $k : \mathbb{N} \to \mathbb{N}$ mit $k(y) = 0$ darf nicht mit der Konstanten 0 gleichgesetzt werden. Zeigen Sie durch syntaktisch korrekte Anwendung der Aufbauregeln für primitive Rekursion, dass $k(y)$ primitiv rekursiv ist.

Sie dürfen annehmen, dass das Schema $f(0) = g(),\ f(m+1) = h(f(m), m)$ ein Spezialfall des Schemas $f(0, \overline{x}) = g(\overline{x}),\ f(m+1, \overline{x}) = h(f(m, \overline{x}), m, \overline{x})$ ist.

Lösungsvorschlag

Um den Blick für den Text der Definition primitiver Funktionen zu schärfen, nehmen wir es dieses Mal sehr genau mit der Beachtung der definierenden Regeln.

Zunächst gibt es keine Regel für eine Definition von $k(n)$ in der Form $k(x) = 0$. Diese Gleichung ist nicht ohne Weiteres ein Spezialfall der Kompositionsregel.

Der folgende Weg ist korrekt.

$$\begin{aligned} h(0) &= 0\,, \\ h(n+1) &= \pi_1^2(h(n), n)\,. \end{aligned}$$

Hausaufgabe 2 (5 Punkte)

Zeigen Sie die primitive Rekursivität der folgenden Funktionen *noteq* und *Div* unter Beachtung der entsprechenden Hinweise:

1. Das 2-stellige Prädikat $noteq(m, n)$, das einem Paar $(m, n) \in \mathbb{N} \times \mathbb{N}$ den Wert 1 zuordnet, falls $m \neq n$ gilt, und andernfalls den Wert 0 besitzt.

2. Die Funktion $Div(m, n)$, die für $n = 0$ den Wert 0 hat und andernfalls durch ganzzahlige Division definiert ist, wie folgt:

$$Div(m, n) = \left\lfloor \frac{m}{n} \right\rfloor .$$

Sie dürfen zusätzlich zu den Basisfunktionen der primitiven Rekursion die erweiterte Komposition und das erweiterte rekursive Definitionsschema benützen. Die in der Vorlesung behandelten primitiv rekursiven Funktionen dürfen Sie ebenfalls benützen.
LOOP-Programme sind nicht erlaubt.

Lösungsvorschlag

1. $noteq(n,m) = 1 \dot{-} (1 \dot{-} ((n \dot{-} m) + (m \dot{-} n)))$.

2. Funktionsverlauf:
   Falls $n \cdot Div(m,n) = m$, dann ist m ohne Rest durch n dividierbar, d. h., dann ist m ein Vielfaches von n. Entsprechend gilt $Div(m+1,n) - Div(m,n) = 1$, wenn $n \cdot Div(m+1,n) = m+1$ gilt, andernfalls gilt $Div(m+1,n) - Div(m,n) = 0$.

   Umrechnung: $n \cdot Div(m+1,n) = m+1$ genau dann, wenn $n \cdot Div(m,n) + n = m+1$.

   In freier Notation:

$$\begin{aligned} Div(0,n) &= 0\,, \\ Div(m+1,n) &= \begin{cases} 0 & : n = 0\,, \\ \left(\begin{cases} Div(m,n) & : noteq(n \cdot Div(m,n) + n, m+1), \\ Div(m,n)+1 & : \text{sonst}\,. \end{cases} \right) & : \text{sonst}\,. \end{cases} \end{aligned}$$

   Mit $ifthen$-Hilfsfunktion:

$$\begin{aligned} &Div(0,n) = 0\,, \\ &Div(m+1,n) = ifthen(n, ifthen[noteq(n \cdot Div(m,n) + n, m+1), \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad Div(m,n), \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad Div(m,n)+1], \quad 0)\,. \end{aligned}$$

Hausaufgabe 3 (5 Punkte)

Zeigen Sie mit den gleichen Hilfsmitteln wie in der vorausgehenden Aufgabe: Falls das Prädikat $P(x,y,z)$ primitiv rekursiv ist, dann ist auch der beschränkte max-Operator

$$Q(m,n) := \max\{k \leq m \mid P(m,n,k)\}$$

primitiv rekursiv.

Lösungsvorschlag

Wie in der Vorlesung setzen wir $\max \emptyset = 0$, wobei man inkaufnimmt, dass der Wert $Q(0,n) = 0$ nichts darüber aussagt, ob $P(0,n,0)$ gilt.

Es gelten dann die folgenden Gleichungen in mathematischer Notation.

$$\begin{aligned} Q(0,n) &= 0\,, \\ Q(m+1,n) &= \begin{cases} m+1 & : P(m+1,n,m+1) \\ Q(m,n) & : \text{ sonst} \end{cases} \end{aligned}$$

Unter der Annahme, dass die modifizierte Subtraktion $\dot{-}$ primitiv rekursiv ist, und mit Verwendung des erweiterten rekursiven Schemas definieren wir

$$\begin{aligned} Q(0,n) &= 0 \\ Q(m+1,n) &= Q(m,n) + \hat{P}(m+1,n,m+1) \cdot ((m+1) \dot{-} Q(m,n))\,. \end{aligned}$$

Hausaufgabe 4 (5 Punkte)

Sei $f : \mathbb{N} \to \mathbb{N}$ diejenige Funktion, die definiert ist durch die Startwerte $f(0) = 1$, $f(1) = 2$, $f(2) = 1$ zusammen mit der Rekursion

$$f(n+1) = f(n-1)^2 f(n-2)^2 \quad \text{für alle } n \in \mathbb{N} \text{ mit } n \geq 2\,.$$

Zeigen Sie, dass f primitiv-rekursiv ist.

Lösungsvorschlag

Wie in Tutoraufgabe 3 von Blatt 9 können 2 alternative Lösungen gegeben werden. Wir geben für f ein erweitertes Schema für primitive Rekursion an.
Analog wie in TA 3, Bl. 9 schreiben wir die Rekursion in Tupel- bzw. Vektorform und suchen eine Funktion f, so dass $(f(0), f(1), f(2))^T = (1, 2, 1)^T$ und für alle $n \in \mathbb{N}$ die folgende Gleichung gilt.

$$\begin{pmatrix} f(n+1) \\ f(n+2) \\ f(n+3) \end{pmatrix} = \begin{pmatrix} f(n+1) \\ f(n+2) \\ f(n+1)^2 f(n)^2 \end{pmatrix}$$

Wir kodieren die 3-Tupel mit Hilfe der primitiv rekursiven Paarungsfunktion $c(x, y)$ zusammen mit den Projektionen p_1, p_2 der Umkehrfunktion von c und definieren $c_3 : \mathbb{N}^3 \to \mathbb{N}$ mit

$$c_3(x, y, z) = c(x, c(y, z))\,.$$

Es gilt $x = p_1(c_3(x, y, z))$, $y = p_1 p_2(c_3(x, y, z))$ und $z = p_2 p_2(c_3(x, y, z))$.
Sei $g : \mathbb{N} \to \mathbb{N}$ mit

$$g(n) = c_3(f(n), f(n+1), f(n+2))\,.$$

Dann erfüllt g das erweiterte Rekursionschema

$$\begin{aligned} g(0) &= c_3(1, 2, 1)\,, \\ g(n+1) &= c_3(\, p_1 p_2(g(n))\,,\, p_2 p_2(g(n))\,,\, p_1 p_2(g(n)) \cdot p_1 p_2(g(n)) \cdot p_1(g(n)) \cdot p_1(g(n))\,)\,. \end{aligned}$$

Da die Multiplikation primitiv rekursiv ist, folgt die primitive Rekursivität von g. Dass f primitiv rekusiv ist, folgt nun aus

$$f(n) = p_1(g(n))\,.$$

PR von f folgt alternativ auch aus einem *LOOP*-Programm.

Vorbereitung 1

Im Folgenden bezeichne $a(n, m)$ die Ackermann-Funktion.

1. Berechnen Sie $a(1, 6)$ und $a(2, 1)$.
2. Zeigen Sie für alle $n, m \in \mathbb{N}$:
   (i) $a(1, m) = m + 2\,,$ (ii) $m < a(n, m)\,.$

Lösungsvorschlag

1. (i) Die Rekursionsgleichungen liefern

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

   (ii) Wir rechnen mit Rekursionsgleichungen 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. (i) $a(1, m) = m + 2$ für alle $m \in \mathbb{N}$:
   Wir beweisen per Induktion über m.
   Für $m = 0$ gilt einerseits $a(1, m) = a(1, 0) = a(0, 1) = 2$. Andrerseits gilt $m + 2 = 2$. Es folgt der Induktionsanfang $a(1, m) = m + 2$ für $m = 0$.
   Der Induktionsschritt von m auf $m + 1$ für $m \geq 0$ folgt aus

$$a(1, m + 1) = a(0, a(1, m)) = a(0, m + 2) = m + 3 = (m + 1) + 2\,.$$

(i) $m < a(n, m)$ für alle n, m:
Wir beweisen per Induktion über n.
Für $n = 0$ gilt die Behauptung wegen $a(0, m) = m + 1 > m$ für alle m.
Induktionsschritt von n auf $n + 1$ für alle $n \geq 0$:
Mit der Induktionsannahme folgt $a(n + 1, 0) = a(n, 1) > 1$. Für alle $m > 0$ gilt zunächst $a(n + 1, m) = a(n, a(n + 1, m - 1)) > a(n + 1, m - 1)$. Damit folgt aber für alle $m > 0$

$$a(n + 1, m) > a(n + 1, m - 1) > a(n + 1, m - 2) > \ldots > a(n + 1, 0) > 1\,.$$

Daraus folgt $a(n + 1, m) > m$.

Vorbereitung 2

Welche Aussagen sind wahr? Geben Sie jeweils eine knappe Begründung an.

1. Jede unentscheidbare Sprache enthält eine entscheidbare Teilmenge.
2. Jede Teilmenge einer entscheidbaren Sprache ist entscheidbar.
3. Für jede unentscheidbare Sprache A gibt es eine echte Obermenge, die ebenfalls unentscheidbar ist.
4. Aus 'A entscheidbar' und '$A \cap B$ entscheidbar' folgt 'B entscheidbar'.

Lösungsvorschlag

Eine Menge $A \subseteq \Sigma^*$ heißt entscheidbar, falls die charakteristische Funktion χ_A total auf Σ^* und berechenbar ist.
Dann ergeben sich die folgenden Antworten und Begründungen.

1. Ja, denn jede Sprache $L \subseteq \Sigma^*$ enthält eine endliche Teilmenge, und jede endliche Teilmenge von Σ^* ist entscheidbar.

   Bemerkung: Eine z. B. einelementige Menge $\{a\}$ ist zwar entscheidbar. Dies muss aber nicht bedeuten, dass man dieses eine Element $a \in \Sigma^*$ kennt bzw. zu konstruieren in der Lage ist. Wir wissen lediglich die abstrakte Existenz dieses a bzw. die Existenz eines Algorithmus, der für vorgelegtes w den Wert der charakteristischen Funktion berechnet, d. h. prüft, ob $w = a$ gilt, wobei der Vergleich $w = a$ für Wörter natürlich mit abbrechendem Algorithmus berechenbar ist.

2. Nein, denn $\{0, 1\}^*$ ist entscheidbar, aber das allgemeine Halteproblem $H \subseteq \{0, 1\}^*$ ist nicht entscheidbar.

3. Ja, denn für ein $w \notin A$ (und das existiert immer!) ist $A \cup \{w\}$ ebenfalls unentscheidbar, falls A unentscheidbar ist.

4. Nein. Gegenbeispiel: $\{a\}$ und $\{a\} \cap H$ sind entscheidbar, weil beide Mengen endlich sind. H (allg. Halteproblem) ist aber nicht entscheidbar.

Vorbereitung 3

1. Falls A auf B mit Funktion f reduzierbar ist, dann gilt $f^{-1}(B) = A$, aber nicht notwendigerweise $f(A) = B$. Beweis!

2. Falls A reduzierbar auf B und B semi-entscheidbar ist, dann ist auch A semi-entscheidbar. Beweis!

3. Sei $B \subseteq \Sigma^*$ mit $B \neq \Sigma^*$ und $B \neq \emptyset$ entscheidbar.
   Zeigen Sie: B ist reduzierbar auf $\Sigma^* \setminus B$.

Lösungsvorschlag

1. Nach Definition gilt $x \in A \Leftrightarrow f(x) \in B$ und wir erinnern an die Definition $f^{-1}(B) = \{x \mid f(x) \in B\}$.

   $x \in A \Rightarrow f(x) \in B$ ist gleichbedeutend mit $f(A) \subseteq B$, woraus insbesondere auch $A \subseteq f^{-1}(B)$ folgt. Aus $x \in A \Leftarrow f(x) \in B$ folgt nun $A \supseteq f^{-1}(B)$. Damit hat man $A = f^{-1}(B)$ bewiesen.

   Durch Angabe eines Beispiels zeigen wir nun, dass i. A. $f(A) \neq B$ gilt.

   Sei $\Sigma^* = \{0,1\}^*$, $B = \{0\}^* \subseteq \Sigma^*$ und $A = \{0x \mid x \in \Sigma^*\}$.

   Die Abbildung $f : \Sigma^* \to \Sigma^*$ mit

   $$f(w) = \begin{cases} 0, & w \in A \\ 1, & w \notin A \end{cases}$$

   ist offenbar total, berechenbar und reduziert A auf B. Es gilt $00 \in B$, aber $00 \notin f(A)$.

   Man bemerkt, dass B beliebig vergrößert werden kann, solange $B \cap f(\overline{A}) = \emptyset$ erfüllt bleibt. Im Beispiel gilt $f(\overline{A}) = \{1\}$.

2. Sei f eine totale, berechenbare Funktion, die A auf B reduziert. Dann gilt für die charakteristische Funktion χ'_B von B

   $w \in A \Rightarrow f(w) \in B \Rightarrow \chi'_B(f(w)) = 1$ und

   $w \notin A \Rightarrow f(w) \notin B \Rightarrow \chi'_B(f(w)) = \perp$.

   Daraus folgt $\chi'_A(w) = \chi'_B(f(w))$.

   Offenbar ist χ_A berechenbar, mithin ist A semi-entscheidbar.

3. Seien $a \in \Sigma^* \setminus B$ und $b \in B$. Dann definieren wir die Abbildung $f : \Sigma^* \to \Sigma^*$ mit

   $$f(w) = \begin{cases} a, & \chi_B(w) = 1\,, \\ b, & \chi_B(w) = 0\,. \end{cases}$$

   f ist offenbar total, berechenbar und reduziert B auf $\Sigma^* \setminus B$.

Tutoraufgabe 1

Wir wollen mit primitiver Rekursion eine Kodierung von Binärbäumen in natürliche Zahlen programmieren.
Dafür sollen primitiv rekursive Funktionen

$$\begin{aligned} \mathit{leaf} &: \mathbb{N} \to \mathbb{N} & \mathit{branch} &: \mathbb{N} \times \mathbb{N} \to \mathbb{N} \\ \mathit{isleaf} &: \mathbb{N} \to \{0,1\} & \mathit{value} &: \mathbb{N} \to \mathbb{N} \\ \mathit{left} &: \mathbb{N} \to \mathbb{N} & \mathit{right} &: \mathbb{N} \to \mathbb{N} \\ \mathit{nleft} &: \mathbb{N} \times \mathbb{N} \to \mathbb{N} \end{aligned}$$

definiert werden.
Dabei repräsentiert $\mathit{leaf}(n)$ ein Blatt mit dem Wert n, und $\mathit{branch}(s,t)$ einen inneren Knoten mit linkem Teilbaum s und rechtem Teilbaum t. Innere Knoten tragen keine Werte. Bei einem Blatt s soll $\mathit{value}(s)$ den Wert des Blattes liefern. Bei einem inneren Knoten s sollen $\mathit{left}(s)$ und $\mathit{right}(s)$ den linken bzw. rechten Teilbaum zurückgeben. Die Funktion $\mathit{nleft}(n,t)$ soll den Baum zurückgeben, der sich ergibt, wenn man n mal die Funktion left auf t anwendet.

1. Geben Sie mit Hilfe der Paarfunktion c aus der Vorlesung und den passenden Projektionen p_1 und p_2 primitiv rekursive Definitionen für leaf, branch, isleaf, value, left und right, und zeigen Sie, dass folgende Gleichungen erfüllt sind:

$$\begin{aligned} \mathit{isleaf}(\mathit{leaf}(n)) &= 1 & \mathit{isleaf}(\mathit{branch}(s,t)) &= 0 \\ \mathit{value}(\mathit{leaf}(n)) &= n \\ \mathit{left}(\mathit{branch}(s,t)) &= s & \mathit{right}(\mathit{branch}(s,t)) &= t \\ \mathit{nleft}(0,t) &= t & \mathit{nleft}(n+1,t) &= \mathit{left}(\mathit{nleft}(n,t)) \end{aligned}$$

   Gehen Sie strikt nach Definition 4.30 vor, und geben Sie jeweils die Instanz des Schemas der primitiven Rekursion bzw. Komposition an, die Sie verwenden.

2. Definieren Sie eine Funktion $\mathit{leftmost} : \mathbb{N} \to \mathbb{N}$, die für einen Baum den Wert des Blattes ermittelt, das am weitesten links steht. Was ist hier die Schwierigkeit?

Lösungsvorschlag

1. Wir machen uns zunächst klar, dass die m-stelligen konstanten Funktionen $C_n^m : \mathbb{N}^m \to \mathbb{N}$ mit $C_n^m(x_1, \ldots, x_m) = n$ primitiv rekursiv sind, da sie sich aus dem Kompositionsschema (mit $k = 0$) ergeben.

   Nun definieren wir:

   $\mathit{leaf}(n) = c(0,n) = c(C_0^1(n), \pi_1^1(n))$
   (Kompositionsschema: $k = 2, g = c, h_1 = C_0^1, h_2 = \pi_1^1$)

   $\mathit{branch}(s,t) = c(1, c(s,t)) = c(C_1^2(s,t), c(s,t))$
   (Kompositionsschema: $k = 2, g = c, h_1 = C_1^2, h_2 = c$)

   $\mathit{isleaf}(t) = 1 \dot{-} p_1(t) = \mathit{minus}(C_1^1(t), p_1(t))$
   (Kompositionsschema: $k = 2, g = \mathit{minus}, h_1 = C_1^1, h_2 = \pi_1^1$)

$value(t) = p_2(t)$
$(value = p_2)$

$left(t) = p_1(p_2(t))$
(Kompositionsschema: $k = 1, g = p_1, h_1 = p_2$)

$right(t) = p_2(p_2(t))$
(Kompositionsschema: $k = 1, g = p_2, h_1 = p_2$)

$help(s, n, t) = left(s) = left(\pi^3_1(s, n, t))$
(Hilfsfunktion nach Kompositionsschema: $k = 1, g = left, h_1 = \pi^3_1$)

$nleft(0, t) = t = \pi^1_1(t)$
$nleft(n + 1, t) = left(nleft(n, t)) = help(nleft(n, t), n, t)$
(Primitive Rekursion: $g = \pi^1_1, h = help$)

Durch nachrechnen sieht man, dass diese Funktionen die geforderten Gleichungen erfüllen:

$isleaf\,(leaf\,(n)) = 1 \dot{-} p_1(c(0, n)) = 1 \dot{-} 0 = 1$
$isleaf\,(branch(s, t)) = 1 \dot{-} p_1(c(1, c(s, t))) = 1 \dot{-} 1 = 0$
$value(leaf\,(n)) = p_2(c(0, n)) = n$
$left(branch(s, t)) = p_1(p_2(c(1, c(s, t)))) = p_1(c(s, t)) = s$
$right(branch(s, t)) = p_2(p_2(c(1, c(s, t)))) = p_2(c(s, t)) = t$

Die Gleichungen für *nleft* entsprechen bereits der Definition.

2. Das Rekursionsschema, das sich für *leftmost* natürlicherweise ergibt, ist keine primitive Rekursion:

$$leftmost(t) = ifthen(isleaf\,(t), value(t), leftmost(left(t)))$$

Hier baut die Definition von $leftmost(t)$ auf $leftmost(left(t))$ auf, wenn der Baum kein Blatt ist. Hier haben wir also nicht den nötigen Schritt von n auf $n + 1$, und somit ist das Rekursionsschema (auch das erweiterte) nicht anwendbar.

Allerdings ist *leftmost* trotzdem primitiv rekursiv. Dafür macht man sich zunächst klar, dass stets $s < branch(s, t)$ gilt. Damit ist für einen als natürliche Zahl s kodierten Baum, s stets eine obere Schranke für die Tiefe des Baums. Nun können wir die Länge des am weitesten links liegenden Zweigs mit beschränkter Maximierung (und Komposition) ausdrücken:

$$\begin{aligned} leftdepth(s) &= \min\{x \leq s \mid isleaf\,(nleft(x, s))\} \\ &= s \dot{-} \max\{x \leq s \mid isleaf\,(nleft(s \dot{-} x, s))\} \end{aligned}$$

Dann definieren wir mit (erweiterten) Kompositionsschema

$$leftmost(s) = value(nleft(leftdepth(s), s))\,.$$

Tutoraufgabe 2

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

1. Zeigen Sie, dass $f(m, n) := (a(m, n))^2$ nicht primitiv rekursiv ist.
2. Zeigen Sie, dass $g(m, n) := \min(3, a(m, n))$ primitiv rekursiv ist.

Lösungsvorschlag

1. 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:

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

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

2. Hier benötigt man nur ein paar wenige Werte der Ackermann-Funktion. Aus den Gleichungen von Folie 302 erstellt man eine kleine Wertetabelle für die Anfangswerte, solange sie ≤ 3 sind:

a	0	1	2	3
0	1	2	3	4
1	2	3	4	5
2	3	5	7	9
3	5	13	29	61

   Man kann also g z.B. wie folgt definieren:

   $$g(m,n) = ifthen(2 \dot{-} (m+n), 3, m+n+1)\ .$$

   Somit ist g lediglich eine Komposition primitiv rekursiver Funktionen.

Tutoraufgabe 3

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

1. $H_{\Sigma^*} = \{w \mid M_w \text{ hält für alle Eingaben}\}$.
2. $H_I = \{w \mid M_w \text{ berechnet die Identitätsfunktion}\}$.
3. Sei $L = \{a^n b^n \mid n \in \mathbb{N}\}$. Dann ist $A = \{w \mid L(M_w) = L\}$ unentscheidbar.

Lösungsvorschlag

1. Wir gehen davon aus (siehe Vorlesung), 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 H_0 auf leerem Band nicht entscheidbar ist.

   Wir reduzieren nun 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:

1. Zunächst wird die Turingmaschine M_w bei leerer Eingabe simuliert (beispielsweise auf einem zweiten Band).

2. Falls M_w hält, dann hält auch $M_{w'}$.

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

2. Wir verfahren analog zur Lösung der vorausgehenden Aufgabe und reduzieren mit Hilfe einer Funktion f das Problem H_0 auf H_I.

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

   1. Zunächst wird die Turingmaschine M_w bei leerer Eingabe simuliert.

   2. Falls M_w hält, dann schreibt $M_{w'}$ die Eingabe y aufs Band.

   Offenbar gilt wiederum $w \in H_0 \iff f(w) \in H_I$, d. h. f reduziert H_0 auf H_I.

3. Sei T_L eine Turingmaschine, die genau L akzeptiert. Wir reduzieren das Problem H_0 auf A analog wie in den vorausgehenden Lösungen.

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

   1. Zunächst wird die Turingmaschine M_w bei leerer Eingabe simuliert.

   2. Falls M_w hält, dann wird die Ausführung von T_L auf y simuliert.

   Offenbar gilt wiederum $w \in H_0 \iff f(w) \in A$, d. h. f reduziert H_0 auf A.