
9. Beispiele primitiv rekursiver Funktionen

Die berechenbaren Funktionen über \mathbb{N} , die man üblicherweise in der Arithmetik betrachtet, lassen sich alle als primitiv rekursiv nachweisen. In diesem Abschnitt werden wir dies für einige Beispiele tun. Hiermit werden wir einerseits die Mächtigkeit des Konzeptes der primitiv rekursiven Funktionen demonstrieren. Andererseits werden die hier erzielten Ergebnisse jedoch auch ein wesentliches Hilfsmittel für den Beweis des Äquivalenzsatzes im nächsten Abschnitt sein. Dies gilt insbesondere für die primitiv rekursive Kodierung endlicher Zahlenfolgen, die wir im letzten Teil dieses Abschnitts betrachten.

Wir beginnen mit der Beobachtung, dass die Klasse der primitiv rekursiven Funktionen gegen explizite Definitionen abgeschlossen ist, d.h. dass jede Funktion, die durch einen Ausdruck (Term) definiert werden kann, der neben den Eingabervariablen (oder einem Teil hiervon) und Zahlenkonstanten nur primitiv rekursive Funktionen enthält, selbst wieder primitiv rekursiv ist. Hierzu benötigen wir zunächst folgende Definitionen.

9.1 DEFINITION. Seien $g_1^{(n_1)}, \dots, g_k^{(n_k)}$ Funktionszeichen der angegebenen Stelligkeiten, sei $V = \{v_1, v_2, v_3, \dots\}$ eine abzählbar unendliche Menge von Variablen, und sei $\underline{\mathbb{N}} = \{\underline{n} : n \in \mathbb{N}\}$ die Menge der Ziffern. Die Menge $T(\underline{g}_1, \dots, \underline{g}_k)$ der *Terme über $\underline{g}_1, \dots, \underline{g}_k$* ist induktiv definiert durch:

- (i) Jede Variable v_n und jede Ziffer \underline{n} ist ein Term über $\underline{g}_1, \dots, \underline{g}_k$.
- (ii) Sind t_1, \dots, t_{n_m} Terme über $\underline{g}_1, \dots, \underline{g}_k$ wobei $1 \leq m \leq k$, so ist auch $\underline{g}_m(t_1, \dots, t_{n_m})$ ein Term über $\underline{g}_1, \dots, \underline{g}_k$.

Ein Term t , der keine Variablen enthält, heißt *geschlossener Term*.

Durch Definition 9.1 wird die syntaktische Gestalt der Terme über gewissen Funktionszeichen festgelegt.

9.2 BEISPIEL. Sind z.B. $\underline{\pm}^{(2)}$ und $\underline{*}^{(2)}$ 2-stellige Funktionszeichen, so ist

$$t = \underline{\pm}(\underline{*}(\underline{2}, \underline{*}(v_1, v_1)), \underline{\pm}(\underline{*}(\underline{3}, v_1), \underline{4}))$$

ein Term über $\underline{\pm}$ und $\underline{*}$. Dies sieht man induktiv unter Verwendung folgender Teilterme (deren Korrektheit wir durch Angabe der verwendeten Klausel aus Definition 9.1 belegen):

(1) $\underline{2}$	(i)
(2) v_1	(i)
(3) $\underline{3}$	(i)
(4) $\underline{4}$	(i)
(5) $\underline{*}(v_1, v_1)$	(ii) : (2), (2)
(6) $\underline{*}(\underline{3}, v_1)$	(ii) : (3), (2)
(7) $\underline{*}(\underline{2}, \underline{*}(v_1, v_1))$	(ii) : (1), (5)
(8) $\underline{+}(\underline{*}(\underline{3}, v_1), \underline{4})$	(ii) : (6), (4)
(9) t	(ii) : (7), (8)

Interpretiert man Variablen als Zahlvariablen, Ziffern \underline{n} als die korrespondierenden Zahlen n und die Funktionszeichen $\underline{g}^{(n)}$ als Funktionen $g^{(n)}$ über \mathbb{N} der entsprechenden Stelligkeit, so stellt ein geschlossener Term eine Zahl und ein Term mit n verschiedenen Variablen eine n -stellige Funktion dar. Diese Semantik der Terme wird in den folgenden Definitionen formal eingeführt. Zunächst ordnen wir jedem geschlossenen Term t seinen Wert $|t|$ zu:

9.3 DEFINITION. Sei $t \in T(\underline{g}_1^{(n_1)}, \dots, \underline{g}_k^{(n_k)})$ ein geschlossener Term und seien $g_1^{(n_1)}, \dots, g_k^{(n_k)}$ Funktionen über \mathbb{N} . Der Wert $|t| = |t|_{g_1, \dots, g_k}$ von t bzgl. g_1, \dots, g_k ist induktiv definiert durch:

1. $|\underline{n}| = n$.
2. $|\underline{g}_i(t_1, \dots, t_{n_i})| = g_i(|t_1|, \dots, |t_{n_i}|)$.

Um die Definition auf Terme mit Variablen zu erweitern, betrachten wir zunächst den Term $t_{w_1, \dots, w_m}[\underline{n}_1, \dots, \underline{n}_m]$, der aus t durch Ersetzen der Variablen w_i durch die Ziffern \underline{n}_i entsteht, sowie die Menge $V(t)$, der in t vorkommenden Variablen.

9.4 DEFINITION. Sei $t \in T(\underline{g}_1^{(n_1)}, \dots, \underline{g}_k^{(n_k)})$, seien $w_1, \dots, w_m \in V$ paarweise verschiedene Variablen und seien $\underline{p}_1, \dots, \underline{p}_m \in \mathbb{N}$.

(a) Der Term

$$t_{\vec{w}}[\vec{p}] = t_{w_1, \dots, w_m}[\underline{p}_1, \dots, \underline{p}_m]$$

ist induktiv definiert durch

1. Für $t \in \mathbb{N}$ ist $t_{\vec{w}}[\vec{p}] = t$.
2. Für $t = v \in V$ ist $t_{\vec{w}}[\vec{p}] = \underline{p}_i$, falls $v = w_i$ ($1 \leq i \leq m$), und $t_{\vec{w}}[\vec{p}] = t$, falls $v \notin \{w_1, \dots, w_m\}$.
3. Für $t = g_i(t_1, \dots, t_{n_i})$ ist $t_{\vec{w}}[\vec{p}] = g_i((t_1)_{\vec{w}}[\vec{p}], \dots, (t_{n_i})_{\vec{w}}[\vec{p}])$.

(b) Die Menge $V(t)$ ist induktiv definiert durch

1. Für $t \in V$ ist $V(t) = \{t\}$.
2. Für $t \in \mathbb{N}$ ist $V(t) = \emptyset$.
3. Für $t = g_i(t_1, \dots, t_{n_i})$ ist $V(t) = V(t_1) \cup \dots \cup V(t_{n_i})$.

9.5 DEFINITION. Sei $t \in T(\underline{g}_1^{(n_1)}, \dots, \underline{g}_k^{(n_k)})$ ein Term mit nichtleerer Variablenmenge $V(t) = \{w_1, \dots, w_m\}$ ($\|V(t)\| = m \geq 1$) und seien $g_1^{(n_1)}, \dots, g_k^{(n_k)}$ Funktionen über \mathbb{N} . Die von t (bzgl. g_1, \dots, g_k) beschriebene Funktion $f_t : \mathbb{N}^m \rightarrow \mathbb{N}$ ist definiert durch

$$\forall \vec{x} = (x_1, \dots, x_m) \in \mathbb{N}^m (f_t(\vec{x}) = |t_{\vec{w}}[\underline{x}_1, \dots, \underline{x}_m]|)$$

9.6 BEISPIEL. Interpretieren wir $\underline{+}^{(2)}$ und $\underline{*}^{(2)}$ als Addition und Multiplikation, so gilt für den in Beispiel 9.2 betrachteten Term t über $\underline{+}, \underline{*}$, dass f_t die 1-stellige Funktion

$$f_t(x) = +(* (2, *(x, x)), +(* (3, x), 4))$$

ist, also in üblicher Infixschreibweise das Polynom

$$f_t(x) = 2 * x * x + 3 * x + 4 = 2x^2 + 3x + 4.$$

In obigen Definitionen können wir auch partielle Funktionen g_1, \dots, g_k zugrundelegen. Der Wert $|t|$ eines geschlossenen Terms kann dann (in offensichtlicher Weise) undefiniert sein und entsprechend wird die von einem Term definierte Funktion i.a. partiell sein. Wir können nun präzisieren, was es bedeutet, dass eine Funktion f explizit über Funktionen g_1, \dots, g_k definierbar ist. Intuitiv gilt dies, wenn f im Sinne der vorhergehenden Definition von einem Term t beschrieben wird, wobei jedoch nur ein Teil der Eingaben in dem Term vorkommen muss. D.h. der Wert von f braucht von gewissen Eingaben nicht abhängen.

9.7 DEFINITION. Die (partielle) Funktion $f^{(n)} : \mathbb{N}^n \rightarrow \mathbb{N}$ ist *explizit definierbar über* den (partiellen) Funktionen $g_1^{(n_1)}, \dots, g_k^{(n_k)}$, wenn es einen Term $t \in T(\underline{g}_1^{(n_1)}, \dots, \underline{g}_k^{(n_k)})$ gibt mit $V(t) = \{w_1, \dots, w_m\}$, wobei $\|V(t)\| = m \leq n$, und es Zahlen $i_1, \dots, i_m \in \{1, \dots, n\}$ gibt mit

$$\forall \vec{x} = (x_1, \dots, x_n) \in \mathbb{N} \quad (f(\vec{x}) = f_t(x_{i_1}, \dots, x_{i_m}) = |t_{w_1, \dots, w_m}[\underline{x}_{i_1}, \dots, \underline{x}_{i_m}]|). \quad (9.1)$$

Liest man hier die Eingaben x_1, \dots, x_n als Variablen, so heißt dies intuitiv, dass man f durch einen Ausdruck darstellen kann, in dem neben den Funktionen g_1, \dots, g_k und Zahlen höchstens diese Variablen vorkommen.

9.8 BEISPIEL. Die Funktion

$$f(x, y) = 2y^2 + 3y + 4$$

wird explizit durch den Term t aus Beispiel 9.2 definiert, wenn wir für die Eingaben $x_1 = x$ und $x_2 = y$ und die Variable $v_1 = w_1$ in Definition 9.7 $i_1 = 2$ wählen, d.h.

$$f(x_1, x_2) = f_t(x_2) = |t_{v_1}[\underline{x}_2]|.$$

9.9 SATZ. F(PRIM) und F(REK) sind gegen explizite Definitionen abgeschlossen.

BEWEIS. Sei $F = F(\text{PRIM})$ oder $F = F(\text{REK})$, seien $g_1^{(n_1)}, \dots, g_k^{(n_k)} \in F$, und sei f gemäß (9.1) gegeben. Zu zeigen: $f \in F$. Wir zeigen dies durch Induktion nach dem Aufbau des Termes t , der f definiert.

1. $t = \underline{m} \in \mathbb{N}$. Dann ist $f(\vec{x}) = |t| = m$ für alle \vec{x} , d.h. $f = C_m^n \in F$.
2. $t \in V$. Dann gilt $V(t) = \{w_1\} = \{t\}$ und $f(\vec{x}) = |t_{w_1}[\underline{x}_{i_1}]| = x_{i_1}$, d.h. $f = U_{i_1}^n \in F$.
3. $t = \underline{g}_i^{(n_i)}(t_1, \dots, t_{n_i})$. Nach I.V. gilt dann $f_l \in F$ für die durch t_l ($1 \leq l \leq n_i$) bzgl. (9.1) definierten Funktionen und es gilt

$$\begin{aligned} f(\vec{x}) &= |t_{\vec{w}}[\underline{x}_{i_1}, \dots, \underline{x}_{i_m}]| \\ &= g_i(|(t_1)_{\vec{w}}[\underline{x}_{i_1}, \dots, \underline{x}_{i_m}]|, \dots, |(t_{n_i})_{\vec{w}}[\underline{x}_{i_1}, \dots, \underline{x}_{i_m}]|) \\ &= g_i(f_1(\vec{x}), \dots, f_{n_i}(\vec{x})) \\ &= g_i(f_1, \dots, f_{n_i})(\vec{x}), \end{aligned}$$

weshalb $f \in F$, da F gegen simultane Substitution abgeschlossen ist.

□

Im Folgenden werden wir den Abschluss von $F(\text{PRIM})$ und $F(\text{REK})$ unter expliziten Definitionen stillschweigend verwenden. Hierbei werden wir die definierenden Terme nicht formal angeben und auch häufig – soweit üblich – die Infixschreibweise verwenden. So sollte z.B. klar sein, dass

$$p(x) = 2x^2 + 3x + 4$$

eine explizite Definition des Polynoms p über $+$ und $*$ ist, weshalb $p \in F(\text{PRIM})$ gilt. (Die Übersetzung der rechten Seite in Präfixschreibweise (unter Ausschreiben von $x^2 = *(x, x)$ und Hinzufügen der weggelassenen Klammern) in den Term aus t aus Beispiel 9.2 werden wir also i.a. nicht durchführen.) Ähnlich werden wir uns bei der Definition einer Funktion $f = \text{PR}(g, h)$ durch primitive Rekursion über Funktionen g und h , die über zuvor schon als primitiv rekursiv nachgewiesenen Funktionen explizit definierbar sind, auf die Angabe der Rekursionsgleichungen beschränken. Es genügt dann, dass die rechte Seite von $f(\vec{x}, 0)$ explizit definierbar ist und ebenso die rechte Seite von $f(\vec{x}, y+1)$, wenn wir dort eventuelle Vorkommen von $f(\vec{x}, y)$ dort durch eine neue Variable z ersetzen.

9.10 SATZ. Die folgenden Funktionen sind primitiv rekursiv:

$$\begin{aligned} f_1(x, y) &= x + y && (\text{Summe}) \\ f_2(x, y) &= x * y && (\text{Produkt}) \\ f_3(x, y) &= x \dot{-} y && (\text{Differenz auf } \mathbb{N}) \\ f_4(x, y) &= |x - y| && (\text{Absolute Differenz}) \\ f_5(x, y) &= \max(x, y) && (\text{Maximum}) \\ f_6(x, y) &= \min(x, y) && (\text{Minimum}) \\ f_7(x) &= \text{sg}(x) && (\text{Signum, Vorzeichen}) \\ & \text{wobei } \text{sg}(x) = \begin{cases} 0 & \text{falls } x = 0 \\ 1 & \text{falls } x > 0 \end{cases} \\ f_8(x) &= \overline{\text{sg}}(x) && (\text{Negiertes Vorzeichen}) \\ & \text{wobei } \overline{\text{sg}}(x) = \begin{cases} 1 & \text{falls } x = 0 \\ 0 & \text{falls } x > 0 \end{cases} \end{aligned}$$

BEWEIS. Dass $+$ und $*$ primitiv rekursiv sind, wurde bereits in Beispiel 8.5 gezeigt. Zum Nachweis der primitiven Rekursivität von $\dot{-}$ zeigt man zunächst, dass $\tilde{f}_3(x) = x \dot{-} 1$ wegen

$$\begin{aligned} 0 \dot{-} 1 &= 0 \\ (x+1) \dot{-} 1 &= x \end{aligned}$$

primitiv rekursiv ist (da man \tilde{f}_3 durch primitive Rekursion aus den über 0 explizit definierbaren Funktionen $g = 0$ und $h(x, z) = x$ erhält), und definiert dann $\dot{-}$ durch primitive Rekursion

$$\begin{aligned} x \dot{-} 0 &= x \\ x \dot{-} (y+1) &= (x \dot{-} y) \dot{-} 1 \end{aligned}$$

(wobei h explizit über $\tilde{f}_3(x) = x \dot{-} 1$ definierbar ist). Die übrigen Funktionen kann man direkt explizit über $+$ und $\dot{-}$ definieren:

$$\begin{aligned}
|x-y| &= (x \dot{-} y) + (y \dot{-} x) \\
\max(x, y) &= x + (y \dot{-} x) \\
\min(x, y) &= \max(x, y) \dot{-} |x-y| \\
\overline{sg}(x) &= 1 \dot{-} x \\
sg(x) &= 1 \dot{-} \overline{sg}(x)
\end{aligned}$$

□

Summe, Produkt, Maximum und Minimum kann man wie folgt verallgemeinern. Zu einer (partiellen) Funktion $g^{(n+1)}$ definiert man:

$$\begin{aligned}
\sigma(g)(\vec{x}, u, o) &= \sum_{i=u}^o g(\vec{x}, i) \\
\sigma_{<}(g)(\vec{x}, y) &= \sum_{i<y} g(\vec{x}, i) \\
\pi(g)(\vec{x}, u, o) &= \prod_{i=u}^o g(\vec{x}, i) \\
\pi_{<}(g)(\vec{x}, y) &= \prod_{i<y} g(\vec{x}, i) \\
\max(g)(\vec{x}, y) &= \max\{g(\vec{x}, i) : i < y\} \\
\min(g)(\vec{x}, y) &= \min\{g(\vec{x}, i) : i < y\}
\end{aligned}$$

(Hierbei vereinbart man, dass

$$\sum_{i<0} \dots = \sum_{i=u}^o \dots = 0 \text{ und } \prod_{i<0} \dots = \prod_{i=u}^o \dots = 1$$

für $o < u$, sowie $\max \emptyset = \min \emptyset = 0$. Bei partiellem g sind die neu definierten Funktionen an den Stellen definiert, die von dem Wert von g nur an solchen Stellen abhängen, an denen g definiert ist. D.h. insbesondere

$$\sigma_{<}(g)(\vec{x}, y) \downarrow \Leftrightarrow \pi_{<}(g)(\vec{x}, y) \downarrow \Leftrightarrow \max(g)(\vec{x}, y) \downarrow \Leftrightarrow \min(g)(\vec{x}, y) \downarrow \Leftrightarrow \forall i < y (g(\vec{x}, i) \downarrow)$$

9.11 SATZ. $F(\text{PRIM})$ und $F(\text{REK})$ sind gegen (die gerade definierte) beschränkte Summenbildung, beschränkte Produktbildung, beschränkte Maximumbildung und beschränkte Minimumbildung abgeschlossen.

BEWEIS. Wir zeigen die Behauptung für die Summenbildung und für $F = F(\text{PRIM})$ und überlassen die anderen Fälle als Übung. Sei also $g^{(n+1)} \in F(\text{PRIM})$. Dann gilt

$$\begin{aligned}
\sigma_{<}(g)(\vec{x}, 0) &= 0 \\
\sigma_{<}(g)(\vec{x}, y+1) &= \sigma_{<}(g)(\vec{x}, y) + g(\vec{x}, y),
\end{aligned}$$

also $\sigma_{<}(g) \in F(\text{PRIM})$ (da $F(\text{PRIM})$ gegen die primitive Rekursion und gegen explizite Definitionen abgeschlossen ist). $\sigma(g) \in F(\text{PRIM})$ folgt hieraus mit der expliziten Definition

$$\sigma(g)(\vec{x}, u, o) = \sigma_{<}(g)(\vec{x}, o+1) \dot{-} \sigma_{<}(g)(\vec{x}, u).$$

□

Ähnlich zeigt man den Abschluss von $F(\text{PRIM})$ und $F(\text{REK})$ gegen Iteration:

9.12 SATZ. $F(\text{PRIM})$ und $F(\text{REK})$ sind gegen Iteration abgeschlossen. D.h. für $f^{(1)} \in F(\text{PRIM})$ (bzw. $F(\text{REK})$) gilt $\text{Iter}(f) \in F(\text{PRIM})$ (bzw. $F(\text{REK})$), wobei $\text{Iter}(f)(x, n) = f^n(x)$.

BEWEIS. Es gilt

$$\begin{aligned}\text{Iter}(f)(x, 0) &= x \\ \text{Iter}(f)(x, n+1) &= f(\text{Iter}(f)(x, n)).\end{aligned}$$

(Für partielles f beachte man, dass $\text{Iter}(f)(x, n) \downarrow$ g. d. w. $f^m(x) \downarrow$ für $1 \leq m \leq n$.) \square

Als nächstes führen wir die primitiv rekursiven und rekursiven Mengen ein. Wir lassen uns dabei von der Beobachtung leiten, dass eine Menge genau dann entscheidbar ist, wenn ihre charakteristische Funktion berechenbar ist (s. Satz 2.10).

9.13 DEFINITION. Eine Menge $M \subseteq \mathbb{N}^n$ ist (*primitiv*) *rekursiv* g. d. w. die charakteristische Funktion c_M von M (*primitiv*) rekursiv ist.

Wir schreiben $M \in \text{F}(\text{PRIM})$ bzw. $M \in \text{F}(\text{REK})$, falls M primitiv rekursiv bzw. rekursiv ist. Mehrstellige Mengen nennen wir auch *Relationen*. Ferner identifizieren wir Mengen mit *Prädikaten*. D.h. eine Eigenschaft P (von natürlichen Zahlen) identifizieren wir mit der Menge der Zahlen, die diese Eigenschaft haben. Also

$$P(\vec{x}) \text{ („}P \text{ trifft auf } \vec{x} \text{ zu“)} \Leftrightarrow \vec{x} \in P.$$

Durch diese Identifizierung entsprechen sich aussagenlogische Operationen (Junktoren) und elementare mengentheoretische Operationen:

Prädikate	Mengen
$\neg P$ (Negation: P trifft <i>nicht</i> zu)	\bar{P} (Komplement)
$P \vee Q$ (Disjunktion: P <i>oder</i> Q trifft zu)	$P \cup Q$ (Vereinigung)
$P \& Q$ (Konjunktion: P <i>und</i> Q treffen zu)	$P \cap Q$ (Durchschnitt)

Man kann jeden aussagenlogischen Junktoren mittels der oben genannten Junktoren darstellen (s. Vorlesungen „Mathematische Logik“ oder „Informatik II“). Man könnte nach den DeMorganschen Regeln sogar auf \vee oder $\&$ verzichten, da

$$\begin{aligned}P \vee Q &\Leftrightarrow \neg(\neg P \& \neg Q) \\ P \& Q &\Leftrightarrow \neg(\neg P \vee \neg Q)\end{aligned}$$

gelten, d.h.

$$P \cup Q = \overline{\bar{P} \cap \bar{Q}} \text{ und } P \cap Q = \overline{\bar{P} \cup \bar{Q}}.$$

Hier betrachten wir als zusätzlichen Junktoren nur noch die *Implikation*

$$P \Rightarrow Q : \Leftrightarrow \neg P \vee Q.$$

Wir wollen nun zeigen, dass $\text{F}(\text{PRIM})$ (und $\text{F}(\text{REK})$) gegen die aussagenlogischen Operationen abgeschlossen sind.

9.14 SATZ. $\text{F}(\text{PRIM})$ und $\text{F}(\text{REK})$ sind gegen \neg , \vee und $\&$ abgeschlossen.

BEWEIS. Für $P, Q \subseteq \mathbb{N}^n$ und $\vec{x} \in \mathbb{N}^n$ gilt:

$$\begin{aligned} c_{\neg P}(\vec{x}) &= \overline{sg}(c_P(\vec{x})) \\ c_{P \vee Q}(\vec{x}) &= \max(c_P(\vec{x}), c_Q(\vec{x})) \\ c_{P \& Q}(\vec{x}) &= \min(c_P(\vec{x}), c_Q(\vec{x})) \end{aligned}$$

□

9.15 KOROLLAR. Die Relationen $=, <, \leq, >, \geq, \neq, \dots$ sind primitiv rekursiv.

BEWEIS. Es gilt $c_{\leq}(x, y) = sg((y+1) - x)$, weshalb $\leq \in F(\text{PRIM})$. Die übrigen Relationen erhält man aus \leq durch Anwendung der aussagenlogischen Operationen:

$$\begin{aligned} x = y &\Leftrightarrow x \leq y \ \& \ y \leq x \\ x \neq y &\Leftrightarrow \neg(x = y) \\ x < y &\Leftrightarrow x \leq y \ \& \ x \neq y \end{aligned}$$

u.s.w.

□

Wie bei den Funktionen sind die (primitiv) rekursiven Mengen gegen explizite Definitionen abgeschlossen, da die (primitive) Rekursivität einer Menge ja über die (primitive) Rekursivität ihrer charakteristischen Funktion definiert ist. Insbesondere gilt:

9.16 LEMMA. Die Klasse der (primitiv) rekursiven Prädikate ist gegen Einsetzung totaler (primitiv) rekursiver Funktionen abgeschlossen. D.h. für (primitiv) rekursives $P \subseteq \mathbb{N}^m$ und totale (primitiv) rekursive Funktionen $f_1^{(n)}, \dots, f_m^{(n)}$ ist das Prädikat $Q \subseteq \mathbb{N}^n$ mit

$$Q(\vec{x}) \Leftrightarrow P(f_1(\vec{x}), \dots, f_m(\vec{x}))$$

wiederum (primitiv) rekursiv.

BEWEIS. Es gilt $c_Q = c_P(f_1, \dots, f_m)$.

□

Die beiden letzten Lemmata zeigen, dass der Graph einer totalen (primitiv) rekursiven Funktion ebenfalls (primitiv) rekursiv ist. Im Fall der Rekursivität gilt auch die Umkehrung. (Im primitiv rekursiven Fall gilt dies nicht, wie später in den Übungen gezeigt werden wird.)

9.17 SATZ. Sei $f: \mathbb{N} \rightarrow \mathbb{N}$ total.

- (a) Die Funktion f ist genau dann rekursiv, wenn der Graph G_f von f rekursiv ist.
 (b) Ist die Funktion f primitiv rekursiv, so ist auch der Graph von f primitiv rekursiv.

BEWEIS. Sei f (primitiv) rekursiv. Dann erhält man den Graphen G_f von f durch Einsetzen von f in das Gleichheitsprädikat:

$$(\vec{x}, y) \in G_f \iff f(\vec{x}) = y.$$

Die primitive Rekursivität von G_f folgt daher aus Korollar 9.15 mit Lemma 9.16. Ist umgekehrt der Graph G_f von f rekursiv, so besitzt f die Darstellung

$$f(\vec{x}) = \mu y ((\vec{x}, y) \in G_f) = \mu y (\overline{sg}(c_{G_f}))(\vec{x}).$$

Da die charakteristische Funktion c_{G_f} des Graphen von f nach Annahme rekursiv ist, folgt hieraus die Rekursivität von f mit Hilfe von Satz 9.10. \square

Da nach der Church-Turing-These (und dem noch zu beweisenden Äquivalenzsatz) die rekursiven Funktionen und Mengen gerade den berechenbaren Funktionen bzw. entscheidbaren Mengen entsprechen, formalisiert Satz 9.17 den entsprechenden Satz 2.14 für die intuitiven Konzepte.

Eine weitere wichtige Abschlusseigenschaft der (primitiv) rekursiven Mengen ist der Abschluss gegen beschränkte Quantoren: Für $P \subseteq \mathbb{N}^{n+1}$ entstehen $P_{\exists}, P_{\forall} \subseteq \mathbb{N}^{n+1}$ aus P durch Anwendung des *beschränkten Existenz-* bzw. *Allquantors*, wobei

$$\begin{aligned} P_{\exists}(\vec{x}, y) &: \Leftrightarrow \exists z < y (P(\vec{x}, z)) &: \Leftrightarrow \exists z (z < y \& P(\vec{x}, z)) \\ P_{\forall}(\vec{x}, y) &: \Leftrightarrow \forall z < y (P(\vec{x}, z)) &: \Leftrightarrow \forall z (z < y \Rightarrow P(\vec{x}, z)) \end{aligned}$$

9.18 SATZ. $F(\text{PRIM})$ und $F(\text{REK})$ sind gegen den beschränkten Existenz- und Allquantor abgeschlossen.

BEWEIS. Sei $F = F(\text{PRIM})$ bzw. $F = F(\text{REK})$ und sei $P \subseteq \mathbb{N}^{n+1}$ mit $c_P \in F$. Dann gilt

$$\begin{aligned} P_{\exists}(\vec{x}, y) &\Leftrightarrow \exists z < y (P(\vec{x}, z)) \\ &\Leftrightarrow \exists z < y (c_P(\vec{x}, z) = 1) \\ &\Leftrightarrow \sum_{z < y} c_P(\vec{x}, z) \geq 1 \\ &\Leftrightarrow \sigma_{<}(c_P)(\vec{x}, y) \geq 1 \end{aligned}$$

weshalb $P_{\exists} \in F$. Weiter kann man den (beschränkten) Allquantor mit Hilfe der Negation durch den (beschränkten) Existenzquantor ausdrücken

$$\begin{aligned} P_{\forall}(\vec{x}, y) &\Leftrightarrow \forall z < y (P(\vec{x}, z)) \\ &\Leftrightarrow \neg \exists z < y (\neg P(\vec{x}, z)) \\ &\Leftrightarrow \neg (\neg P)_{\exists}(\vec{x}, z) \end{aligned}$$

weshalb auch $P_{\forall} \in F$. \square

Wegen Lemma 9.16 ist mit $P(\vec{x}, y)$ nicht nur $\exists y < z (P(\vec{x}, y))$ wiederum primitiv rekursiv, sondern auch $\exists y < f(z) (P(\vec{x}, y))$ für jedes $f \in F(\text{PRIM})$. Insbesondere ist also auch (mit $f = S$) $\exists y \leq z (P(\vec{x}, y))$ primitiv rekursiv. (Analog für den beschränkten Allquantor.)

Allgemein stellen die bislang gezeigten Abschlusseigenschaften der primitiv rekursiven Mengen sicher, dass jedes Prädikat, das explizit definiert ist über primitive rekursive Prädikate, die durch die aussagenlogischen Junktoren verknüpft sind, beschränkt quantifiziert sein dürfen, und in die primitiv rekursive Funktion eingesetzt sein können, wiederum primitiv rekursiv ist.

9.19 BEISPIEL. Die Menge $PZ \subseteq \mathbb{N}$ der Primzahlen ist wegen der Darstellung

$$PZ(n) \Leftrightarrow n \geq 2 \& \forall x < n (\forall y < n (x \cdot y \neq n))$$

primitiv rekursiv.

Der μ -Operator, der ja die kleinste Nullstelle einer Funktion berechnet, kann auf beliebige Prädikate sinngemäß wie folgt übertragen werden: Für $P \subseteq \mathbb{N}^{n+1}$ ist $f^{(n)} = \mu(P)$ die partielle Funktion, die bei Eingabe \vec{x} das kleinste y mit $P(\vec{x}, y)$ ausgibt und undefiniert ist, falls solch ein y nicht existiert. D.h. aber gerade, dass $\mu(P)(\vec{x}) = \mu(c_{\overline{P}})(\vec{x})$, weshalb – für rekursives P – $\mu(P) \in \text{F(REK)}$ gilt.

Die Klasse F(PRIM) ist gegen den folgenden *beschränkten μ -Operator* abgeschlossen:

$$\mu_b(P)(\vec{x}, y) = \begin{cases} \mu z < y (P(\vec{x}, z)) & \text{falls } \exists z < y (P(\vec{x}, z)) \\ 0 & \text{sonst} \end{cases}$$

(Im folgenden werden wir entsprechend $\mu z < y (P(\vec{x}, z))$ als 0 lesen, falls es kein $z < y$ mit $P(\vec{x}, z)$ gibt.)

9.20 SATZ. F(PRIM) ist gegen den beschränkten μ -Operator abgeschlossen.

BEWEIS. Zu $P \subseteq \mathbb{N}^n$ aus F(PRIM) definieren wir zunächst

$$Q(\vec{x}, y) \Leftrightarrow P(\vec{x}, y) \ \& \ \forall z < y (\neg P(\vec{x}, z))$$

Dann ist $Q \in \text{F(PRIM)}$ und $Q(\vec{x}, y)$ gilt genau für das kleinste y mit $P(\vec{x}, y)$ (falls P überhaupt auf ein (\vec{x}, y) zutrifft). Es folgt hieraus, dass

$$\begin{aligned} \mu_b(P)(\vec{x}, y) &= \mu z < y (P(\vec{x}, z)) \\ &= \mu z < y (Q(\vec{x}, z)) \\ &= \sum_{z < y} (z \cdot c_Q(\vec{x}, z)) \end{aligned}$$

weshalb $\mu_b(P) \in \text{F(PRIM)}$. □

Wir setzen unsere Untersuchung der Abschlusseigenschaften von F(PRIM) und F(REK) mit dem Nachweis fort, dass diese Klassen F gegen endliche Fallunterscheidungen abgeschlossen sind. D.h. sind P_1, \dots, P_k n -stellige Prädikate aus F , die \mathbb{N}^n zerlegen (d.h. $P_1 \cup \dots \cup P_k = \mathbb{N}^n$ und P_1, \dots, P_k sind paarweise disjunkt), und sind $f_1^{(n)}, \dots, f_k^{(n)}$ (partielle) Funktionen aus F , so ist auch

$$f(\vec{x}) = \begin{cases} f_1(\vec{x}) & \text{falls } P_1(\vec{x}) \\ \vdots \\ f_k(\vec{x}) & \text{falls } P_k(\vec{x}) \end{cases}$$

wiederum in F .

9.21 SATZ. F(REK) und F(PRIM) sind gegen endliche Fallunterscheidungen abgeschlossen.

BEWEIS. Sei $\text{F} = \text{F(REK)}$ bzw. $\text{F} = \text{F(PRIM)}$, seien $f_1, \dots, f_k, P_1, \dots, P_k \in \text{F}$ und sei f hieraus wie oben definiert. Weiter nehmen wir an, dass die Funktionen f_1, \dots, f_k alle total sind (was bei $\text{F} = \text{F(PRIM)}$ natürlich stets der Fall ist). Dann gilt

$$f(\vec{x}) = \sum_{i=1}^k f_i(\vec{x}) \cdot c_{P_i}(\vec{x}),$$

da für das durch \vec{x} eindeutig festgelegte i mit $P_i(\vec{x})$ der zugehörige Summand $f_i(\vec{x}) \cdot c_{P_i}(\vec{x}) = f_i(\vec{x})$ ist, während die anderen Summanden (wegen $c_{P_j}(\vec{x}) = 0$) verschwinden. Aus dieser Darstellung von f folgt $f \in F$.

Für partielle f_i ist obige Darstellung nicht notwendigerweise korrekt. Gilt z.B. $f_1(\vec{x}) \downarrow$ und $P_1(\vec{x})$, während $f_2(\vec{x}) \uparrow$, so gilt $f(\vec{x}) = f_1(\vec{x}) \downarrow$, während

$$\sum_{i=1}^k f_i(\vec{x}) \cdot c_{P_i}(\vec{x}) \uparrow,$$

da aus $f_2(\vec{x}) \uparrow$ folgt, dass auch der Summand $f_2(\vec{x}) \cdot c_{P_2}(\vec{x}) = *(f_2, c_{P_2})(\vec{x})$ undefiniert ist (nach Definition der simultanen Substitution) und damit auch die Summe insgesamt. Bei partiellen f_1, \dots, f_k muss man daher anders vorgehen:

Zu einer beliebigen gegebenen partiell rekursiven Funktion $g^{(m)}$ und einem rekursiven Prädikat $P^{(n)}$ definiert man zunächst eine $(m+n)$ -stellige partielle Funktion $g_P \in F(\text{REK})$ mit $(\vec{x} \in \mathbb{N}^m, \vec{y} \in \mathbb{N}^n)$

$$g_P(\vec{x}, \vec{y}) = \begin{cases} g(\vec{x}) & \text{falls } P(\vec{y}) \\ 0 & \text{sonst.} \end{cases}$$

Dieses g_P erhält man durch Induktion nach dem Aufbau von g (s. Übungen). Dann kann man f mit Hilfe der derart definierten Varianten $(f_i)_{P_i}$ von f_i (die für \vec{x} mit $\neg P_i(\vec{x})$ stets 0 sind) durch

$$f(\vec{x}) = \sum_{i=1}^k ((f_i)_{P_i}(\vec{x}, \vec{x}) \cdot c_{P_i}(\vec{x}))$$

definieren. □

Wir nennen $f^{(n)}$ eine *endliche Variante* von $f^{(n)}$ und schreiben $f \stackrel{*}{=} f'$, wenn sich f und f' nur in endlich vielen Argumenten unterscheiden. Entsprechend ist M *endliche Variante* von M' , $M \stackrel{*}{=} M'$, falls $c_M \stackrel{*}{=} c_{M'}$.

9.22 SATZ. *Jede endliche Menge ist primitiv rekursiv. $F(\text{PRIM})$ und $F(\text{REK})$ sind gegen endliche Varianten abgeschlossen.*

BEWEIS FÜR $F(\text{PRIM})$. Die (n) -stellige leere Menge $\emptyset^{(n)}$ ist primitiv rekursiv, da $c_{\emptyset^{(n)}} = C_0^n$. Für nicht leeres endliches $M = \{\vec{x}_1, \dots, \vec{x}_n\}$ gilt

$$M(\vec{x}) \Leftrightarrow \vec{x} = \vec{x}_1 \vee \dots \vee \vec{x} = \vec{x}_n$$

woraus $M \in F(\text{PRIM})$ folgt.

Seien nun $f^{(n)}, f'^{(n)}$ totale Funktionen mit $f \stackrel{*}{=} f'$ und $f \in F(\text{PRIM})$. Dann ist $Df(f, f') = \{\vec{x} : f(\vec{x}) \neq f'(\vec{x})\}$ endlich und ohne Einschränkung nicht leer, etwa $Df(f, f') = \{\vec{x}_1, \dots, \vec{x}_k\}$. Man kann also f' durch die endliche Fallunterscheidung

$$f'(\vec{x}) = \begin{cases} f'(\vec{x}_1) & \text{falls } \vec{x} = \vec{x}_1 \\ \vdots & \vdots \\ f'(\vec{x}_k) & \text{falls } \vec{x} = \vec{x}_k \\ f(\vec{x}) & \text{sonst} \end{cases}$$

definieren. Die Prädikate P_1, \dots, P_{k+1} zur Unterscheidung der Fälle sind hierbei endlich bzw. co-endlich, also, nach dem gerade Gezeigten, primitiv rekursiv. Dasselbe gilt für die Funktionen f_1, \dots, f_{k+1} , die f' in den einzelnen Fällen beschreiben, da diese entweder konstant oder – im Fall von f_{k+1} – die als primitiv rekursiv vorausgesetzte Funktion f sind. Es ist also nach Satz 9.21 die Funktion f' ebenfalls primitiv rekursiv. \square

Im Rest dieses Paragraphen geben wir primitive rekursive Kodierungen von endlichen Zahlenfolgen an. Hierzu betrachten wir zunächst Paare von Zahlen, dann Zahlenfolgen beliebiger aber fester Länge, und schließlich beliebige endliche Zahlenfolgen.

Man erhält eine bijektive Abbildung $\tau : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$, d.h. eine Nummerierung der Zahlenpaare, indem man – anschaulich gesprochen – den rechten oberen Quadranten der (diskreten) Zahlenebene vollständig durchläuft, wobei man alle Nebendiagonalen der Reihe nach von (rechts) unten nach (links) oben durchläuft (Diagramm: s. Vorlesung). Die erhaltene Funktion τ kann man auch explizit durch

$$\tau(x, y) = \frac{1}{2}(x^2 + 2xy + y^2 + x + 3y) = ((x + y)^2 + (x + y) + 2y) \quad (9.2)$$

definieren.

9.23 SATZ. Für die durch (9.2) definierte Funktion $\tau : \mathbb{N}^2 \rightarrow \mathbb{N}$ gilt:

- (i) τ ist bijektiv.
- (ii) τ ist monoton, d.h. für x_1, x'_1, x_2, x'_2 mit $x_1 \leq x'_1$ und $x_2 \leq x'_2$ gilt $\tau(x_1, x_2) \leq \tau(x'_1, x'_2)$. Ferner gilt $x_1, x_2 \leq \tau(x_1, x_2)$.
- (iii) τ ist primitiv rekursiv.
- (iv) Die zugehörigen Projektionsfunktionen π_1 und π_2 mit $\pi_i(\tau(x_1, x_2)) = x_i$ ($i = 1, 2$) sind ebenfalls primitiv rekursiv.

BEWEIS. Wir verzichten auf den Beweis von (i) (s. Übung). Teil (ii) folgt unmittelbar aus (9.2) und der Monotonie der Addition und Multiplikation. (iii) folgt ebenfalls direkt aus der Definition (9.2), da diese eine explizite Definition von τ über den primitiv rekursiven Funktionen $+$, $*$, div darstellt. (Dabei ist div die ganzzahlige Division auf \mathbb{N} , von der die primitive Rekursivität in den Übungen gezeigt wird.) Zum Beweis von (iv) genügt es, wegen der oben gezeigten Abschlusseigenschaften von $F(\text{PRIM})$ die Projektion π_1 (und analog π_2) durch

$$\pi_1(x) = \mu x_1 \leq x (\exists x_2 \leq x (\tau(x_1, x_2) = x))$$

zu charakterisieren. \square

Durch Iteration von τ können wir n -Tupel von Zahlen, d.h. Folgen fester Länge über \mathbb{N} kodieren:

9.24 DEFINITION. Die Funktionen $\tau_n : \mathbb{N}^n \rightarrow \mathbb{N}$ ($n \geq 1$) sind induktiv definiert durch

$$\begin{aligned} \tau_1(x_1) &= x_1 \\ \tau_{n+1}(x_1, x_2, \dots, x_{n+1}) &= \tau(x_1, \tau_n(x_2, \dots, x_{n+1})) \end{aligned}$$

9.25 SATZ. Für $n \geq 1$ gilt:

- (i) $\tau_n : \mathbb{N}^n \rightarrow \mathbb{N}$ ist bijektiv.
- (ii) τ_n ist monoton und $x_1, \dots, x_n \leq \tau_n(x_1 \dots x_n)$.
- (iii) τ_n ist primitiv rekursiv.
- (iv) Die zu τ_n gehörenden Projektionsfunktionen $\pi_i^n : \mathbb{N} \rightarrow \mathbb{N}$ mit

$$\pi_i^n(\tau_n(x_1, \dots, x_n)) = x_i \quad (1 \leq i \leq n)$$

sind primitiv rekursiv.

BEWEIS. Der Beweis wird durch Induktion nach n geführt. Der Fall $n = 1$ ist dabei trivial. Sei also $n + 1 > 1$ gegeben und gelte der Satz nach I.V. für $1, \dots, n$. Da $\tau_{n+1}(x_1, \dots, x_{n+1}) = \tau(x_1, \tau_n(x_2, \dots, x_{n+2}))$ folgen (i) – (iii) einfach aus der Tatsache, dass die entsprechenden Eigenschaften für τ (nach Satz 9.22) und τ_n (nach I.V.) gelten. Zum Beweis von (iv) beobachtet man, dass wegen (ii)

$$\begin{aligned} \pi_i^{n+1}(x) = \mu x_i \leq x (\exists x_1 \leq x \dots \exists x_{i-1} \leq x \exists x_{i+1} \leq x \dots \\ \dots \exists x_{n+1} \leq x (\tau_{n+1}(x_1, \dots, x_{n+1}) = x)) \end{aligned}$$

weshalb $\pi_i^{n+1} \in F(\text{PRIM})$. □

Die Möglichkeit, Zahlenfolgen fester Länge primitiv rekursiv durch Zahlen zu kodieren, können wir verwenden, um zu zeigen, dass $F(\text{PRIM})$ gegen folgendes Rekursionsschema abgeschlossen ist.

9.26 DEFINITION. Die Funktionen $f_1^{(n+1)}, \dots, f_k^{(n+1)}$ ($k \geq 1$) entstehen aus $g_1^{(n)}, \dots, g_k^{(n)}$ und $h_1^{(n+1+k)}, \dots, h_k^{(n+1+k)}$ durch *simultane primitive Rekursion*, wenn (für $1 \leq i \leq k$)

$$\begin{aligned} f_i(\vec{x}, 0) &= g_i(\vec{x}) \\ f_i(\vec{x}, y+1) &= h_i(\vec{x}, y, f_1(\vec{x}, y), \dots, f_k(\vec{x}, y)) \end{aligned}$$

(Schreibweise: $(f_1, \dots, f_k) = \text{SPR}(g_1, \dots, g_k; h_1, \dots, h_k)$.)

Hier werden also mehrere Funktionen gleichzeitig durch Rekursion nach der letzten Variable y definiert, wobei der Wert einer Funktion an der Stelle $y+1$ von den Werten *aller* Funktionen an der vorherigen Stelle y abhängen darf.

9.27 SATZ. $F(\text{PRIM})$ und $F(\text{REK})$ sind gegen *simultane primitive Rekursion* abgeschlossen.

BEWEIS (FÜR $F(\text{PRIM})$). Gelte $(f_1, \dots, f_k) = \text{SPR}(g_1, \dots, g_k; h_1, \dots, h_k)$ für $g_1^{(n)}, \dots, g_k^{(n)}, h_1^{(n+1+k)}, \dots, h_k^{(n+1+k)} \in F(\text{PRIM})$. Um $f_1, \dots, f_k \in F(\text{PRIM})$ nachzuweisen, zeigen wir mit Hilfe einer einfachen primitiven Rekursion, dass

$$f(\vec{x}, y) = \tau_k(f_1(\vec{x}, y), \dots, f_k(\vec{x}, y))$$

primitiv rekursiv ist, woraus dann die Behauptung mit $f_i(\vec{x}, y) = \pi_i^k(f(\vec{x}, y))$ folgt:

$$\begin{aligned} f(\vec{x}, 0) &= \tau_k(g_1(\vec{x}), \dots, g_k(\vec{x})) \\ f(\vec{x}, y+1) &= \tau_k(h_1(\vec{x}, y, \pi_1^k(f(\vec{x}, y))), \dots, \pi_k^k(f(\vec{x}, y))), \\ &\quad \dots, h_k(\vec{x}, y, \pi_1^k(f(\vec{x}, y))), \dots, \pi_k^k(f(\vec{x}, y))) \end{aligned}$$

□

Abschließend können wir mit Hilfe der Funktionen τ_n nun endliche Zahlenfolgen variabler Länge kodieren.

9.28 DEFINITION. Sei \mathbb{N}^* die Menge aller endlichen Folgen über \mathbb{N} . Die Funktion $\tau^* : \mathbb{N}^* \rightarrow \mathbb{N}$ ist definiert durch:

$$\begin{aligned} \tau^*(\lambda) &= 0 \\ \tau^*(x_1, \dots, x_n) &= \tau(n-1, \tau_n(x_1, \dots, x_n)) + 1 \end{aligned}$$

9.29 LEMMA. $\tau^* : \mathbb{N}^* \rightarrow \mathbb{N}$ ist eine Bijektion.

BEWEIS. Dies ergibt sich leicht aus der Bijektivität der Funktionen τ und τ_n für $n \geq 1$.

□

Da τ^* auf \mathbb{N}^* und nicht auf \mathbb{N}^n definiert ist, kann τ^* im Gegensatz zu den Funktionen τ und τ_n nicht primitiv rekursiv sein. Wir können jedoch zeigen, dass wir alle wichtigen Informationen über die Folge x_1, \dots, x_n aus der Zahl $\tau^*(x_1, \dots, x_n)$ auf primitiv rekursive Weise dekodieren können:

9.30 SATZ. Die wie folgt definierten 1-stelligen Funktionen l , head , tail und 2-stelligen Funktionen π^* , \circ sind primitiv rekursiv:

$$l(\tau^*(\vec{x})) = |\vec{x}|$$

$$\text{head}(\tau^*(\vec{x})) = \begin{cases} x_1 & \text{falls } \vec{x} = (x_1, \dots, x_n) \\ 0 & \text{sonst} \end{cases}$$

$$\text{tail}(\tau^*(\vec{x})) = \begin{cases} \tau^*(x_2, \dots, x_n) & \text{falls } \vec{x} = (x_1, \dots, x_n) \text{ mit } n \geq 2 \\ \tau^*(\lambda) (= 0) & \text{sonst} \end{cases}$$

$$\pi^*(\tau^*(\vec{x}), i) = \begin{cases} x_i & \text{falls } \vec{x} = (x_1, \dots, x_n) \text{ mit } n \geq i \geq 1 \\ 0 & \text{sonst} \end{cases}$$

$$\tau^*(\vec{x}) \circ \tau^*(\vec{y}) = \tau^*(\vec{x}, \vec{y})$$

Für die durch x kodierte Folge \vec{x} gibt also $l(x)$ die Länge von \vec{x} , $\text{head}(x)$ die erste Komponente x_1 von \vec{x} und $\pi^*(x, i)$ allgemein die i -te Komponente x_i von \vec{x} an, während $\text{tail}(x)$ die Restfolge von \vec{x} nach Streichen von x_1 ist. Entsprechend kodiert \circ die Konkatenation.

BEWEIS. Dass $l, \text{head}, \text{tail} \in \text{F}(\text{PRIM})$ sind, ergibt sich aus

$$l(x) = \begin{cases} 0 & \text{falls } x = 0 \\ \pi_1(x \dot{-} 1) + 1 & \text{sonst} \end{cases}$$

$$\text{head}(x) = \begin{cases} 0 & \text{falls } l(x) = 0 \\ \pi_2(x \dot{-} 1) & \text{falls } l(x) = 1 \\ \pi_1(\pi_2(x \dot{-} 1)) & \text{sonst} \end{cases}$$

(Man beachte, dass für $x = \tau^*(x_1)$

$$\pi_2(x \dot{-} 1) = \tau_1(x_1) = x_1$$

während für $x = \tau^*(x_1, \dots, x_n)$ mit $n \geq 2$

$$\pi_2(x \dot{-} 1) = \tau_n(x_1, \dots, x_n) = \tau(x_1, \tau_{n-1}(x_2, \dots, x_n))$$

$$\text{tail}(x) = \begin{cases} 0 & \text{falls } l(x) \leq 1 \\ \tau(l(x) \dot{-} 2, \pi_2(\pi_2(x \dot{-} 1))) + 1 & \text{sonst} \end{cases}$$

(Man beachte, dass für $x = \tau^*(\vec{x})$ und $\vec{x} = (x_1, \dots, x_n)$ mit $n \geq 2$,

$$\pi_2(\pi_2(x \dot{-} 1)) = \tau_{n-1}(x_2, \dots, x_n),$$

woraus sich die Definition von $\text{tail}(x)$ im 2. Fall ergibt, wenn man τ_{n-1} noch nach τ_* übersetzt.) Um $\pi^* \in F(\text{PRIM})$ zu zeigen, zeigen wir zunächst, dass die Funktion $\text{end}(x, y)$ mit

$$\text{end}(\tau^*(\vec{x}), y) = \begin{cases} \tau^*(x_{y+1}, \dots, x_n) & \text{falls } \vec{x} = (x_1, \dots, x_n) \text{ mit } n > y \\ \tau^*(\lambda) = 0 & \text{sonst} \end{cases}$$

primitiv rekursiv ist. Dies ergibt sich aus folgender Rekursion:

$$\begin{aligned} \text{end}(x, 0) &= x \\ \text{end}(x, y + 1) &= \text{tail}(\text{end}(x, y)) \end{aligned}$$

Hiermit lässt sich π^* rekursiv durch

$$\begin{aligned} \pi^*(x, 0) &= 0 \\ \pi^*(x, y + 1) &= \begin{cases} \text{head}(\text{end}(x, y)) & \text{falls } l(\text{end}(x, y)) \geq 1 \\ 0 & \text{sonst} \end{cases} \end{aligned}$$

definieren.

Schließlich zum Nachweis von $\circ \in F(\text{PRIM})$ werden wir zeigen, dass es eine primitiv rekursive Funktion $\sigma(n)$ gibt, sodass für jede Folge $\vec{x} = (x_1, \dots, x_m)$ mit $x_1, \dots, x_m, m \leq n$ gilt, dass $\tau^*(\vec{x}) < \sigma(n)$ ist. Da nach Definition von τ^* auch $x_1, \dots, x_m, m \leq \tau^*(\vec{x})$ gilt, folgt hieraus für beliebige Folgen \vec{x}, \vec{y}

$$\tau^*(\vec{x}) \circ \tau^*(\vec{y}) = \tau^*(\vec{x}, \vec{y}) < \sigma(\tau^*(x) + \tau^*(y)).$$

Dies erlaubt \circ mit Hilfe des beschränkten μ -Operators zu definieren:

$$\begin{aligned} x \circ y &= \mu z < \sigma(x + y) (\forall i < l(x) (\pi^*(z, i + 1) = \pi^*(x, i + 1)) \\ &\quad \& \forall i < l(y) (\pi^*(z, l(x) + i + 1) = \pi^*(y, i + 1))) \end{aligned}$$

Es bleibt $\sigma \in F(\text{PRIM})$ zu zeigen. Hierzu beobachten wir zunächst, dass für die durch

$$\begin{aligned}\sigma_0(n, 0) &= n \\ \sigma_0(n, m+1) &= \tau(n, \sigma_0(n, m))\end{aligned}$$

definierte primitiv rekursive Funktion σ_0 nach Definition von τ_n und wegen der Monotonie von τ und τ_n gilt:

$$\forall \vec{x} = (x_1, \dots, x_p) ([x_1, \dots, x_p \leq n \ \& \ p \leq m+1] \Rightarrow \tau_p(\vec{x}) \leq \sigma_0(n, m))$$

Nach Definition von τ^* können wir also

$$\sigma(n) = \tau(n, \sigma_0(n, n)) + 1$$

wählen. □

Für kodierte Folgen benutzen wir auch folgende Schreibweise:

$$\begin{aligned}\langle x_1, \dots, x_n \rangle &:= \tau^*(x_1, \dots, x_n) \\ \langle \rangle &:= \tau^*(\lambda) \\ (x)_i &:= \pi^*(x, i) \\ (x)_{i,j} &:= \pi^*(\pi^*(x, i), j)\end{aligned}$$

Mit Hilfe der primitiv rekursiven Folgenkodierung können wir zeigen, dass $F(\text{REK})$ und $F(\text{PRIM})$ gegen die Wertverlaufsrekursion abgeschlossen sind (siehe Übungen). Bei dieser darf der Wert von $f(\vec{x}, y+1)$ nicht nur von $f(\vec{x}, y)$, sondern von allen vorhergehenden Werten $f(\vec{x}, 0), \dots, f(\vec{x}, y)$ abhängen.

9.31 DEFINITION. $f^{(n+1)} = \text{WR}(g, h)$ entsteht aus $g^{(n)}$ und $h^{(n+2)}$ durch Wertverlaufsrekursion, falls

$$\begin{aligned}f(\vec{x}, 0) &= g(\vec{x}) \\ f(\vec{x}, y+1) &= h(\vec{x}, y, \tau^*(f(\vec{x}, 0), \dots, f(\vec{x}, y))).\end{aligned}$$

Ein Beispiel für eine durch Wertverlaufsrekursion definierte Funktion ist die Aufzählungsfunktion f der Fibonacci-Zahlen

$$\begin{aligned}f(0) &= f(1) = 1 \\ f(n+2) &= f(n) + f(n+1),\end{aligned}$$

die im Rekursionsschritt nicht nur auf den letzten Wert sondern auf die zwei letzten Werte zurückgreift.

Will man zeigen, dass die Funktion $\text{sum}(x)$ mit

$$\text{sum}(\langle x_1, \dots, x_n \rangle) = \sum_{i=1}^n x_i$$

primitiv rekursiv ist, so kann man dies auch mit Wertverlaufsrekursion tun:

$$\begin{aligned}\text{sum}(0) &= \text{sum}(\langle \rangle) = 0 \\ \text{sum}(\langle x_1, \dots, x_n \rangle) &= \text{sum}(\langle x_1, \dots, x_{n-1} \rangle) + x_n\end{aligned}$$

Hier greift man im Rekursionsschritt zwar nur auf einen vorhergehenden Wert, aber nicht auf den letzten Wert, zurück.