Technische Universität München
Fakultät für Informatik
Dr. H. Ehler, Dr. M. Pizka,
W. Prenninger, Dr. M. Schneider

WS 2001
Lösungsvorschläge zu Blatt 11
20. Dezember 2001

Übungen zu Einführung in die Informatik I

Aufgabe 44 Semantik rekursiver Funktionsdeklarationen (Lösungsvorschlag)

a) $\tau : (\mathbb{N}^{\perp} \times \mathbb{N}^{\perp} \to \mathbb{N}^{\perp}) \to (\mathbb{N}^{\perp} \times \mathbb{N}^{\perp} \to \mathbb{N}^{\perp})$

$$\tau[g](x,y) = \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ x * g(x, y-1), & \text{falls } x \neq \perp \wedge y > 0 \end{cases}$$

b) Gemäß Vorlesung sind die Approximationsfunktionen f_n: $\mathbb{N}^{\perp} \times \mathbb{N}^{\perp} \to \mathbb{N}^{\perp}$; $\quad n \in \mathbb{N}$ wie folgt definiert. Für alle $x, y \in \mathbb{N}^{\perp}$:

$$\begin{array}{rcl} f_0(x,y) & = & \perp \\ f_{n+1}(x,y) & = & \tau[f_n](x,y) \end{array}$$

Wir ermitteln die ersten vier Approximationsfunktionen:

$$\begin{array}{rrl} & f_0(x,y) = & \perp \\ & f_1(x,y) = & \tau[f_0](x,y) \\ \texttt{Def. von } \tau & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ x * f_0(x, y-1), & \text{falls } x \neq \perp \wedge y > 0 \end{cases} \\ \texttt{f}_0\texttt{, s.o.} & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ x * \perp, & \text{falls } x \neq \perp \wedge y > 0 \end{cases} \\ \texttt{Striktheit von *} & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \vee y \geq 1 \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \end{cases} \end{array}$$

$$
\begin{array}{rrl}
f_2(x,y) & = & \tau[f_1](x,y) \\
\texttt{Def. von } \tau & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ x * f_1(x,y-1), & \text{falls } x \neq \perp \wedge y > 0 \end{cases} \\
\texttt{f}_1\texttt{, s.o.} & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ x * \perp, & \text{falls } x \neq \perp \wedge y > 0 \wedge (x = \perp \vee y-1 = \perp \vee y-1 \geq 1) \\ x * 1, & \text{falls } x \neq \perp \wedge y > 0 \wedge x \neq \perp \wedge y-1 = 0 \end{cases} \\
\texttt{* strikt; Vereinfachung} & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ \perp, & \text{falls } x \neq \perp \wedge y > 1 \\ x, & \text{falls } x \neq \perp \wedge y = 1 \end{cases} \\
\texttt{Zus.fsg 1. und 3. Fall} & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \vee y \geq 2 \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ x, & \text{falls } x \neq \perp \wedge y = 1 \end{cases}
\end{array}
$$

$$
\begin{array}{rrl}
f_3(x,y) & = & \tau[f_2](x,y) \\
 & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ x * f_2(x,y-1), & \text{falls } x \neq \perp \wedge y > 0 \end{cases} \\
\texttt{f}_2\texttt{, s.o.} & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ x * \perp, & \text{falls } x \neq \perp \wedge y > 0 \wedge (x = \perp \vee y-1 = \perp \vee y-1 \geq 2) \\ x * 1, & \text{falls } x \neq \perp \wedge y > 0 \wedge x \neq \perp \wedge y-1 = 0 \\ x * x, & \text{falls } x \neq \perp \wedge y > 0 \wedge x \neq \perp \wedge y-1 = 1 \end{cases} \\
\texttt{* strikt; Vereinfachung} & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ \perp, & \text{falls } x \neq \perp \wedge y \geq 3 \\ x, & \text{falls } x \neq \perp \wedge y = 1 \\ x^2, & \text{falls } x \neq \perp \wedge y = 2 \end{cases} \\
\texttt{Zus.fsg 1. und 3. Fall} & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \vee y \geq 3 \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ x, & \text{falls } x \neq \perp \wedge y = 1 \\ x^2, & \text{falls } x \neq \perp \wedge y = 2 \end{cases}
\end{array}
$$

c) Es gilt für alle $n \in \mathbb{N}$ und für alle $x, y \in \mathbb{N}^\perp$:

$$f_n(x,y) = \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \vee y \geq n \\ x^y, & \text{falls } x \neq \perp \wedge y < n \end{cases}$$

Dabei sei 0^0 definiert durch $0^0 = 1$.
Beweis der Formel für f_n durch Indukion über n.

- *Induktionsanfang:* $n = 0$
  Für alle $x, y \in \mathbb{N}^\perp$ gilt:

$$
\begin{array}{lrcl}
\texttt{Teilaufgabe (b)} & f_0(x,y) & = & \perp \\
\texttt{es gilt } \neg(y < 0) & & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \vee y \geq 0 \\ x^y, & \text{falls } x \neq \perp \wedge y < 0 \end{cases}
\end{array}
$$

- *Induktionsschritt:*
  Induktionsannahme: Für alle $x, y \in \mathbb{N}^{\perp}$ gilt:

$$f_n(x,y) = \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \vee y \geq n \\ x^y, & \text{falls } x \neq \perp \wedge y < n \end{cases}$$

zu zeigen: Für alle $x, y \in \mathbb{N}^{\perp}$ gilt:

$$f_{n+1}(x,y) = \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \vee y \geq n+1 \\ x^y, & \text{falls } x \neq \perp \wedge y < n+1 \end{cases}$$

Es gilt für beliebige $x, y \in \mathbb{N}^{\perp}$:

$$\begin{array}{rcl} f_{n+1}(x,y) & = & \tau[f_n](x,y) \\ \texttt{Def. von } \tau & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ x * f_n(x, y-1), & \text{falls } x \neq \perp \wedge y > 0 \end{cases} \\ \texttt{Induktionsannahme} & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ x * \perp, & \text{falls } x \neq \perp \wedge y > 0 \wedge (x = \perp \vee y-1 = \perp \vee y-1 \geq n) \\ x * x^{y-1}, & \text{falls } x \neq \perp \wedge y > 0 \wedge x \neq \perp \wedge y-1 < n \end{cases} \\ \texttt{* strikt} & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \\ 1, & \text{falls } x \neq \perp \wedge y = 0 \\ \perp, & \text{falls } x \neq \perp \wedge y \geq n+1 \\ x^y, & \text{falls } x \neq \perp \wedge y > 0 \wedge y < n+1 \end{cases} \\ \texttt{Zus.fsg 1/3 \& 2/4} & = & \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \vee y \geq n+1 \\ x^y, & \text{falls } x \neq \perp \wedge y < n+1 \end{cases} \end{array}$$

Damit haben wir die Behauptung gezeigt.

d) Es gilt für alle $x, y \in \mathbb{N}^{\perp}$:

$$f^{\infty}(x,y) = \begin{cases} \perp, & \text{falls } x = \perp \vee y = \perp \\ x^y, & \text{falls } x \neq \perp \wedge y \neq \perp \end{cases}$$

e) Wie alle in der Vorlesungsnotation formulierbaren Funktionale ist auch τ stetig. Nach dem Satz von Kleene gilt für den kleinsten Fixpunkt **fix** τ von τ:

$$\mathbf{fix}\tau = f^{\infty}$$

Sei g ein beliebiger Fixpunkt von τ, d.h. gelte $\tau[g] = g$. Da **fix** τ der kleinste Fixpunkt von τ ist, gilt **fix** $\tau \sqsubseteq g$. Nach Definition von $\sqsubseteq$ auf Funktionen ist dies äquivalent zu $(\forall x, y \in \mathbb{N}^{\perp} : \mathbf{fix}\tau(x,y) \sqsubseteq g(x,y))$. Die Ordnung $\sqsubseteq$ auf $\mathbb{N}^{\perp}$ ist flach. Da für alle $x, y \in \mathbb{N}$ gilt **fix** $\tau(x,y) \neq \perp$, folgt für diese x, y : **fix** $\tau(x,y) = g(x,y)$. Da g ein Fixpunkt von τ ist, folgt im Fall $x = \perp \vee y = \perp$: $g(x,y) = \perp$. Also gilt für alle $x, y \in \mathbb{N}^{\perp}$: **fix** $\tau(x,y) = g(x,y)$.
Somit ist **fix** τ der einzige Fixpunkt von τ.

Aufgabe 45 Spezifikation und partielle Korrektheit (Lösungsvorschlag)

a) Spezifikation der Aufgabestellung von istPraefix:

fct istPraefix$_s$ = (**seq m** x, **seq m** y) **bool**: (*)
$\exists$ **seq m** z: x∘z = y

b) rekursive Rechenvorschrift, die die Spezifikation (*) implementiert:

fct istPraefix = (**seq m** x, **seq m** y) **bool**:
  if x = empty **then** true
  elif y = empty **then** false
    else first(x)$\stackrel{?}{=}$first(y) $\wedge$ istPraefix(rest(x), rest(y))

c) Aus der rekursiven Rechenvorschrift istPraefix leiten wir folgendes Funktional ab:
$\tau : ((M^*)^\perp \times (M^*)^\perp \to \mathbb{B}^\perp) \to ((M^*)^\perp \times (M^*)^\perp \to \mathbb{B}^\perp)$,

$$\tau[g](x,y) = \begin{cases} \perp & \text{falls } x = \perp \vee y = \perp \\ \text{true} & \text{falls } x = \text{<>} \wedge y \neq \perp \\ \text{false} & \text{falls } x \neq \perp \wedge x \neq \text{<>} \wedge y = \text{<>} \\ first(x) \stackrel{?}{=} first(y) \wedge g(rest(x), rest(y)) & \text{falls } x \neq \perp \wedge y \neq \perp \wedge \\ & \quad x \neq \text{<>} \wedge y \neq \text{<>} \end{cases}$$

d) Sei f_s eine Funktion, die die Spezifikation (*) erfüllt, dann zeigen wir, dass $f_s = \tau[f_s]$ gilt. Wir unterscheiden folgende Fälle:
1. Fall: Sei $x = \perp$ oder $y = \perp$, dann gilt aufgrund der Definition von τ bzw. der Striktheit von f_s: $\tau[f_s](x,y) = \perp = f_s(x,y)$.
2. Fall: Sei $x, y \in M^*$ mit $x = \text{<>}$, dann gilt: $\tau[f_s](x,y) = true = f_s(x,y)$, denn mit $z = y$ gilt $x \circ y = y$.
3. Fall: Sei $x, y \in M^*$ mit $x \neq \text{<>}$ und $y = \text{<>}$, dann gilt: $\tau[f_s](x,y) = false = f_s(x,y)$, da es kein $z \in M^*$ mit $x \circ z = y$ geben kann.
4. Fall: Sei $x, y \in M^*$ mit $x = \text{<}x_1, \ldots, x_k\text{>}$ und $y = \text{<}y_1, \ldots, y_n\text{>}$ und $k, n \geq 1$, dann gilt:

$$\begin{aligned} \tau[f_s](x,y) &= first(x) \stackrel{?}{=} first(y) \wedge f_s(rest(x), rest(y)) \\ &= x_1 \stackrel{?}{=} y_1 \wedge f_s(\text{<}x_2, \ldots, x_k\text{>}, \text{<}y_2, \ldots, y_n\text{>}) \\ &= f_s(\text{<}x_1, \ldots, x_k\text{>}, \text{<}y_1, \ldots, y_n\text{>}) \\ &= f_s(x,y) \end{aligned}$$

, denn es gibt genau dann ein $z \in M^*$ mit $x \circ z = y$, wenn $x_1 = y_1$ gilt und ein $z' \in M^*$ mit $\text{<}x_2, \ldots, x_k\text{>} \circ z' = \text{<}y_2, \ldots, y_n\text{>}$ existiert.

f_s ist also Fixpunkt von τ. Da mit der rekursiven Rechenvorschrift von istPraefix der kleinste Fixpunkt verbunden ist, gilt:

$$\text{istPraefix} \sqsubseteq f_s\ .$$

Also ist die rekursive Rechenvorschrift istPraefix partiell korrekt bezüglich der Spezifikation (*).

Aufgabe 46 Effizienz rekursiver Algorithmen (Lösungsvorschlag)

a) **fct** perm$_a$ = (**seq m** s) **nat** : 1+perm1$_a$(<>, <>, s),
fct perm1$_a$ = (**seq m** t, **seq m** l, **seq m** r) **nat** :
  if r $\stackrel{?}{=}$<> **then** 1
    elif (rest(r) $\stackrel{?}{=}$<>) $\wedge$ (l $\stackrel{?}{=}$<>) **then** 1
     else 1+perm1$_a$(t $\circ$ <first(r)>, <>, l $\circ$ rest(r)) + perm1$_a$(t, l $\circ$ <first(r)>, rest(r))
    fi
  fi

b) Für die Rechenvorschrift $perm_a$ ergibt sich wegen $perm_a(s) = 1 + perm1_a(<>, <>, s)$ die Abschätzung:

$$1 + |s|! \leq perm_a(s) \leq 2 + |s| * |s|!$$

Nachdem die Ausgabesequenz in jedem Fall $|s|!$ Permutationen von s enthält, ist grundsätzlich nur eine kleine Verringerung der Zahl der Aufrufe von $O(|s| * |s|!)$ auf $O(|s|!)$ denkbar.

Aufgabe 47 Analyse von Funktionen (Lösungsvorschlag)

a) Das gesuchte Funktional τ mit

$$\tau : (\text{String}^{\perp} \rightarrow \text{String}^{\perp}) \rightarrow (\text{String}^{\perp} \rightarrow \text{String}^{\perp})$$

lautet:

$$\tau[f](\text{xs}) = \begin{cases} \perp; & \text{xs} = \perp \\ \text{Empty}; & \text{xs} = \text{Empty} \\ f(\text{rest (xs)}) \circ < \text{first (xs)} >; & \text{xs} \neq \text{Empty} \wedge \text{xs} \neq \perp \end{cases}$$

b) Die Funktionenfolge f_i ist durch

$$\begin{array}{lcll} f_0 & = & \perp & \forall xs \in \text{String}^{\perp} \\ f_{i+1}(xs) & = & \tau[f_i](xs) & \forall xs \in \text{String}^{\perp} \end{array}$$

gegeben. Somit ergibt sich:

$$\begin{array}{lcl} f_0(\text{xs}) & = & \perp \\ f_1(\text{xs}) & = & \tau[f_0](\text{xs}) = \begin{cases} \perp; & \text{xs} = \perp \\ \text{Empty}; & \text{xs} = \text{Empty} \\ \perp; & \text{length (xs)} \geq 1 \end{cases} \\ f_2(\text{xs}) & = & \tau[f_1](\text{xs}) = \begin{cases} \perp; & \text{xs} = \perp \\ \text{Empty}; & \text{xs} = \text{Empty} \\ \text{<first (xs)>}; & \text{length (xs)} = 1 \\ \perp; & \text{length (xs)} \geq 2 \end{cases} \\ f_3(\text{xs}) & = & \tau[f_2](\text{xs}) = \begin{cases} \perp; & \text{xs} = \perp \\ \text{Empty}; & \text{xs} = \text{Empty} \\ \text{<first (xs)>}; & \text{length (xs)} = 1 \\ \text{<first (rest (xs))>} \circ < \text{first (xs)} >; & \text{length (xs)} = 2 \\ \perp; & \text{length (xs)} \geq 3 \end{cases} \end{array}$$

Man erkennt, dass die Funktion f_{i+1}, Textsequenzen der Länge i revertiert. Für Listen mit length(xs) > i ist das Resultat $\perp$. Damit ergibt sich für i > 1:

$$f_{i+1}(\text{xs}) = \tau[f_i](\text{xs}) = \begin{cases} \perp; & \text{xs} = \perp \\ \text{Empty}; & \text{xs} = \text{Empty} \\ \text{<first (xs)>}; & \text{length (xs)} = 1 \\ \text{<first (rest (xs))>} \circ < \text{first (xs)} >; & \text{length (xs)} = 2 \\ \quad\vdots & \\ \quad\vdots & \\ \text{<first (rest}^{i-1}\text{(xs))>} \circ \cdots & \text{length (xs)} = i \\ \quad \cdots \text{<first (rest (xs))>} \circ < \text{first (xs)} >; & \\ \perp; & \text{length (xs)} \geq (i+1) \end{cases}$$

c) Es sei xs $\in$ String$^\perp$. Dann setzen wir die Lösung der Funktionalgleichung

$$\tau[F](xs) = F(xs)$$

mit

$$F(\text{xs}) = \begin{cases} \perp; & \text{xs} = \perp \\ \text{Empty}; & \text{xs} = \text{Empty} \\ \text{<first (rest}^{(length(xs)-1)}\text{(xs))>} \circ \cdots & \text{length (xs)} > 0 \\ \quad \cdots \text{<first (rest (xs))>} \circ < \text{first (xs)} >; & \end{cases}$$

an. Die Korrektheit dieses Ansatzes zeigen wir durch Einsetzen. Nun gilt laut Definition des Operators τ:

$$\tau[F](xs) = \begin{cases} \perp, & \text{xs} = \perp \\ \text{Empty}, & \text{xs} = \text{Empty} \\ \text{F (rest(xs))} \circ \text{<first (xs)>} & \text{length (xs) > 0} \end{cases}$$

Verwenden wir die Definition von F, so ergibt sich:

$$\tau[F](xs) = \begin{cases} \perp, & \text{xs} = \perp \\ \text{Empty}, & \text{xs} = \text{Empty} \\ \perp \circ \text{<first (xs)>} & \text{rest (xs)} = \perp \\ \text{Empty} \circ \text{<first (xs)>} & \text{rest (xs)} = \text{Empty} \\ \text{<first (rest}^{(length(rest(xs))-1)}\text{(rest (xs)))>} \circ \cdots & \text{length (rest (xs)) > 0} \\ \quad \cdots < \text{first (rest (xs))} > \circ < \text{first (xs)} >; & \end{cases}$$

Da

$$\text{<first (rest}^{(length(rest(xs))-1)}\text{(rest (xs)))>} = \text{<first (rest}^{(length(xs)-1)}\text{(xs))>}$$

folgt

$$\tau[F](xs) = \begin{cases} \perp, & \text{xs} = \perp \\ \text{Empty}, & \text{xs} = \text{Empty} \\ \text{<first (rest}^{(length(xs)-1)}\text{(xs))>} \circ \cdots & \text{length (xs) > 0} \\ \quad \cdots < \text{first (rest (xs))} > \circ < \text{first (xs)} >; & \end{cases}$$

und somit $\tau[F](xs) = F(xs)$.

d) Implementieren wir die Funktionen f_0, f_1, f_2 und f_3 sowie das Funktional τ in Gofer so erhalten wir:

```
{- Implementierung des Funktionales tau -}

tau :: (String -> String) -> (String -> String)
tau f [] = []
tau f (x:xs) = f xs ++ [x]

{- Implementierung von f0, f1, f2 und f3 -}

f0 :: String -> String
f0 xs = undefined

f1 :: String -> String
f1 xs = tau f0 xs
```

```
f2 :: String -> String
f2 xs = tau f1 xs

f3 :: String -> String
f3 xs = tau f2 xs

{-
   Implementierung einer Funktion, die einen String
   beliebiger Laenge revertiert unter Verwendung des
   Funktionales tau
-}

rev :: String -> String
rev xs = tau rev xs
```