TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK

Lehrstuhl für Sprachen und Beschreibungsstrukturen WS 2008/09
Einführung in die Informatik 2 **Übungsblatt 12**
Prof. Dr. Helmut Seidl, T. M. Gawlitza, S. Pott, M. Schwarz 13.01.2009

Abgabe: 20.01.2009 (vor der Vorlesung)

Aufgabe 12.1 (H) Verifikation funktionaler Programme

Es seien folgende, bekannte Definitionen gegeben:

```
let rec len = fun l ->
  match l with
    []     -> 0
  | x::xs -> 1 + len xs

let rec app = fun l1 l2 ->
  match l1 with
    []     -> l2
  | x::xs -> x :: app xs l2

let rec sorted = fun l ->
  match l with
    []     -> true
  | x::xs ->
      match xs with
        []    -> true
      | y::_ -> x <= y & sorted xs

let rec insert = fun y l ->
  match l with
    []     -> [y]
  | x::xs ->
      match x <= y with
        true  -> x :: insert y xs
      | false -> y::x::xs

let rec sort = fun l ->
  match l with
    []     -> []
  | [x]    -> [x]
  | x::xs -> insert x (sort xs)
```

a) Zeigen Sie, dass unter der Voraussetzung, dass alle Aufrufe terminieren,

$$\texttt{len}\ (\texttt{app l1 l2}) = \texttt{len l1} + \texttt{len l2}$$

für alle Listen $\texttt{l1}, \texttt{l2}$ gilt.

b) Zeigen Sie, dass unter der Voraussetzung, dass alle Aufrufe terminieren,

$$\text{Aus } \texttt{sorted l} \text{ folgt } \texttt{sorted}\ (\texttt{insert y l}) \tag{1}$$

für alle $\mathtt{y}, \mathtt{l}$ gilt. **Hinweis:** Im Induktionsschritt müssen sie verschiedene Fälle unterscheiden. Bei diesen Fällen kommt es auf die Beziehung zwischen $\mathtt{y}$ und den ersten beiden Elementen der Liste $\mathtt{l}$ an.

c) Zeigen Sie, dass unter der Voraussetzung, dass alle Aufrufe terminieren,

$$\mathtt{sorted}\ (\mathtt{sort}\ \mathtt{xs}) = \mathbf{true}$$

für alle Listen $\mathtt{xs}$ gilt.

Lösungsvorschlag 12.1

a) **Induktionsanfang:** Es gilt $\mathtt{l1} = []$. Dann gilt:

$$\begin{aligned}\mathtt{len}\ (\mathtt{app}\ \mathtt{l1}\ \mathtt{l2}) &= \mathtt{len}\ \mathtt{l2}\\ &= 0 + \mathtt{len}\ \mathtt{l2}\\ &= \mathtt{len}\ \mathtt{l1} + \mathtt{len}\ \mathtt{l2}\end{aligned}$$

Induktionsschluss: Es gilt $\mathtt{l1} = \mathtt{x} :: \mathtt{xs}$. Dann gilt:

$$\begin{aligned}\mathtt{len}\ (\mathtt{app}\ \mathtt{l1}\ \mathtt{l2}) &= \mathtt{len}\ (\mathtt{app}\ (\mathtt{x} :: \mathtt{xs})\ \mathtt{l2})\\ &= \mathtt{len}\ (\mathtt{x} :: \mathtt{app}\ \mathtt{xs}\ \mathtt{l2})\\ &= 1 + \mathtt{len}\ (\mathtt{app}\ \mathtt{xs}\ \mathtt{l2})\\ &= 1 + \mathtt{len}\ \mathtt{xs} + \mathtt{len}\ \mathtt{l2} && \text{(Induktionsannahme)}\\ &= \mathtt{len}\ (\mathtt{x} :: \mathtt{xs}) + \mathtt{len}\ \mathtt{l2}\\ &= \mathtt{len}\ \mathtt{l1} + \mathtt{len}\ \mathtt{l2}\end{aligned}$$

b) Sei also $\mathtt{sorted}\ \mathtt{l}$ erfüllt.

Induktionsanfang: Wir unterscheiden drei Fälle

Fall 1: Es gilt $\mathtt{l} = []$. Dann folgt:

$$\begin{aligned}\mathtt{sorted}\ (\mathtt{insert}\ \mathtt{y}\ \mathtt{l}) &= \mathtt{sorted}\ (\mathtt{insert}\ \mathtt{y}\ [])\\ &= \mathtt{sorted}\ ([\mathtt{y}])\\ &= \mathbf{true}\end{aligned}$$

Fall 2: Es gilt $\mathtt{l} = [\mathtt{x}]$ und $\mathtt{x} \leq \mathtt{y}$. Dann folgt:

$$\begin{aligned}\mathtt{sorted}\ (\mathtt{insert}\ \mathtt{y}\ \mathtt{l}) &= \mathtt{sorted}\ (\mathtt{insert}\ \mathtt{y}\ [\mathtt{x}])\\ &= \mathtt{sorted}\ (\mathtt{x} :: (\mathtt{insert}\ \mathtt{y}\ []))\\ &= \mathtt{sorted}\ (\mathtt{x} :: [\mathtt{y}])\\ &= \mathtt{x} <= \mathtt{y}\ \&\ \mathtt{sorted}\ [\mathtt{y}]\\ &= \mathbf{true}\ \&\ \mathbf{true}\\ &= \mathbf{true}\end{aligned}$$

Fall 3: Es gilt $\mathtt{l} = [\mathtt{x}]$ und $\mathtt{y} < \mathtt{x}$. Dann folgt:

$$\begin{aligned}\mathtt{sorted}\ (\mathtt{insert}\ \mathtt{y}\ \mathtt{l}) &= \mathtt{sorted}\ (\mathtt{insert}\ \mathtt{y}\ (\mathtt{x} :: []))\\ &= \mathtt{sorted}\ (\mathtt{y} :: \mathtt{x} :: [])\\ &= \mathtt{y} <= \mathtt{x}\ \&\ \mathtt{sorted}\ (\mathtt{x} :: [])\\ &= \mathbf{true}\ \&\ \mathtt{sorted}\ \mathtt{l}\\ &= \mathbf{true}\end{aligned}$$

Induktionsschluss: Wir unterscheiden drei Fälle:

Fall 1: Es gilt $\mathtt{l} = \mathtt{x} :: \mathtt{xs}$ und $\mathtt{y} < \mathtt{x}$. Dann folgt:

$$\begin{aligned}
\mathtt{sorted}\ (\mathtt{insert\ y\ l}) &= \mathtt{sorted}\ (\mathtt{insert\ y}\ (\mathtt{x} :: \mathtt{xs})) \\
&= \mathtt{sorted}\ (\mathtt{y} :: \mathtt{x} :: \mathtt{xs}) \\
&= \mathtt{y} <= \mathtt{x}\ \&\ \mathtt{sorted}\ (\mathtt{x} :: \mathtt{xs}) \\
&= \mathbf{true}\ \&\ \mathtt{sorted\ l} \\
&= \mathbf{true}
\end{aligned}$$

Fall 2: Es gilt $\mathtt{l} = \mathtt{x}_1 :: \mathtt{x}_2 :: \mathtt{xs}$ und $\mathtt{x}_1 \leq \mathtt{y} < \mathtt{x}_2$. Dann folgt:

$$\begin{aligned}
&\mathtt{sorted}\ (\mathtt{insert\ y\ l}) \\
&= \mathtt{sorted}\ (\mathtt{insert\ y}\ (\mathtt{x}_1 :: \mathtt{x}_2 :: \mathtt{xs})) \\
&= \mathtt{sorted}\ (\mathtt{x}_1 :: \mathtt{insert\ y}\ (\mathtt{x}_2 :: \mathtt{xs})) \\
&= \mathtt{sorted}\ (\mathtt{x}_1 :: \mathtt{y} :: \mathtt{x}_2 :: \mathtt{xs}) \\
&= \mathtt{x}_1 <= \mathtt{y}\ \&\ \mathtt{sorted}\ (\mathtt{y} :: \mathtt{x}_2 :: \mathtt{xs}) \\
&= \mathtt{x}_1 <= \mathtt{y}\ \&\ \mathtt{y} <= \mathtt{x}_2\ \&\ \mathtt{sorted}\ (\mathtt{x}_2 :: \mathtt{xs}) \\
&= \mathtt{x}_1 <= \mathtt{y}\ \&\ \mathtt{y} <= \mathtt{x}_2\ \&\ \mathtt{sorted}\ (\mathtt{x}_2 :: \mathtt{xs}) \\
&= \mathtt{x}_1 <= \mathtt{y}\ \&\ \mathtt{y} <= \mathtt{x}_2\ \&\ (\mathtt{x}_1 \leq \mathtt{x}_2\ \&\ \mathtt{sorted}\ (\mathtt{x}_2 :: \mathtt{xs})) \qquad (\text{da } \mathtt{x}_1 < \mathtt{x}_2) \\
&= \mathtt{x}_1 <= \mathtt{y}\ \&\ \mathtt{y} <= \mathtt{x}_2\ \&\ \mathtt{sorted}\ (\mathtt{x}_1 :: \mathtt{x}_2 :: \mathtt{xs}) \\
&= \mathtt{x}_1 <= \mathtt{y}\ \&\ \mathtt{y} <= \mathtt{x}_2\ \&\ \mathtt{sorted\ l} \\
&= \mathbf{true}
\end{aligned}$$

Fall 3: Es gilt $\mathtt{l} = \mathtt{x}_1 :: \mathtt{x}_2 :: \mathtt{xs}$ und $\mathtt{x}_1 \leq \mathtt{x}_2 \leq \mathtt{y}$. Dann folgt:

$$\begin{aligned}
&\mathtt{sorted}\ (\mathtt{insert\ y\ l}) \\
&= \mathtt{sorted}\ (\mathtt{insert\ y}\ (\mathtt{x}_1 :: \mathtt{x}_2 :: \mathtt{xs})) \\
&= \mathtt{sorted}\ (\mathtt{x}_1 :: \mathtt{insert\ y}\ (\mathtt{x}_2 :: \mathtt{xs})) \\
&= \mathtt{sorted}\ (\mathtt{x}_1 :: \mathtt{x}_2 :: \mathtt{insert\ y\ xs}) \\
&= \mathtt{x}_1 <= \mathtt{x}_2\ \&\ \mathtt{sorted}\ (\mathtt{x}_2 :: \mathtt{insert\ y\ xs}) \\
&= \mathbf{true}\ \&\ \mathtt{sorted}\ (\mathtt{insert\ y}\ (\mathtt{x}_2 :: \mathtt{xs})) \qquad (\mathtt{x}_2 \leq \mathtt{y}) \\
&= \mathbf{true} \qquad (\text{Induktionsannahme})
\end{aligned}$$

c) **Induktionsanfang:** Falls $\mathtt{l1} = []$ gilt, dann gilt:

$$\begin{aligned}
\mathtt{sorted}\ (\mathtt{sort}\ []) &= \mathtt{sorted}\ [] \\
&= \mathbf{true}
\end{aligned}$$

Falls $\mathtt{l1} = [\mathtt{x}]$ gilt, dann gilt:

$$\begin{aligned}
\mathtt{sorted}\ (\mathtt{sort}\ [\mathtt{x}]) &= \mathtt{sorted}\ [\mathtt{x}] \\
&= \mathbf{true}
\end{aligned}$$

Induktionsschluss: Es gilt $\mathtt{l1} = \mathtt{x} :: \mathtt{xs}$. Dann gilt:

$$\begin{aligned}
\mathtt{sorted}\ (\mathtt{sort\ x} :: \mathtt{xs}) &= \mathtt{sorted}\ (\mathtt{insert\ x}\ (\mathtt{sort\ xs})) \\
&= \mathbf{true} \\
&\qquad (\text{wegen } \mathtt{sorted\ sort\ xs} = \mathbf{true} \text{ und Teil b) })
\end{aligned}$$

Aufgabe 12.2 (P) app und rev

Gegeben sei folgende MiniOcaml-Funktion zur Umkehrung einer Liste

```
let rec rev = fun list ->
  match list with
    [] -> []
  | el :: rest -> app (rev rest) [el]
```

sowie die in der Vorlesung wie folgt definierte Funktion `app`

```
let rec app = fun x -> fun y ->
  match x with
    [] -> y
  | x::xs -> x :: app xs y
```

Zeigen Sie unter der Annahme, dass alle vorkommenden Funktionsaufrufe terminieren, dass folgende Aussagen gelten:

a) $\texttt{rev (app xs ys)} = \texttt{app (rev ys) (rev xs)}$

b) $\texttt{rev (rev list)} = \texttt{list}$

Hinweis: Folgende Aussagen sind bereits in der Vorlesung bewiesen worden:

$$\text{app x (app y z)} = \text{app (app x y) z} \tag{2}$$

$$\text{app x []} = \text{x} \tag{3}$$

Lösungsvorschlag 12.2

Zur Korrektheit unserer Induktionsbeweise benötigen wir, dass sämtliche vorkommenden Funktionsaufrufe terminieren. Im Folgenden setzen wir also Terminierung voraus. (siehe Vorlesung!)

a) • $\texttt{xs}_0 = []$: Gemäß der angegebenen Definition gilt dann $\texttt{rev xs}_0 = []$. Somit folgt:

$$\begin{aligned}\texttt{rev (app xs}_0\texttt{ ys)} &= \texttt{rev (app [] ys)}\\ &= \texttt{rev ys}\\ &\overset{(3)}{=} \texttt{app (rev ys) []}\\ &= \texttt{app (rev ys) (rev xs}_0\texttt{)}\end{aligned}$$

• **Induktionsschluss:** Wir betrachten die Liste $\texttt{xs}' = \texttt{x :: xs}$. Nach Induktionsannahme ist das zu beweisende Prädikat für Listen mit einer Länge $l < length(xs')$ gültig, d.h. die Aussage $\texttt{rev (app xs ys)} = \texttt{app (rev ys) (rev xs)}$ ist gültig. Wir bilden nun $\texttt{rev(app xs}'\texttt{ ys)}$. Dann gilt (unter Verwendung der bisher gezeigten Beziehungen):

$$\begin{aligned}\texttt{rev (app xs}'\texttt{ ys)} &\overset{Def.\ \texttt{app}}{=} \texttt{rev (x :: (app xs ys))}\\ &\overset{Def.\ \texttt{rev}}{=} \texttt{app (rev (app xs ys)) [x]}\\ &\overset{IA}{=} \texttt{app (app (rev ys) (rev xs)) [x]}\\ &\overset{(2)}{=} \texttt{app (rev ys) (app (rev xs) [x])}\\ &\overset{IA}{=} \texttt{app (rev ys) (rev (app [x] xs)))}\\ &\overset{Def.}{=} \texttt{app (rev ys) (rev (x :: xs))}\\ &= \texttt{app (rev ys) (rev xs}'\texttt{)}\end{aligned}$$

b) Zu beweisen ist nun das Prädikat

$$\texttt{rev}\ (\texttt{rev}\ \texttt{xs}) = \texttt{xs}$$

Der Beweis erfolgt mittels Induktion:

Induktionsanfang: $\texttt{xs} = []$. Dann gilt gemäß der Definition der Funktion `rev`:

$$\begin{aligned}
\texttt{rev}\ (\texttt{rev}\ \texttt{xs}) &= \texttt{rev}\ (\texttt{rev}\ []) \\
&= \texttt{rev}\ [] \\
&= []
\end{aligned}$$

Induktionsschluss: Es wird gezeigt, dass aus der Gültigkeit von $\texttt{rev}\ (\texttt{rev}\ \texttt{xs}) = \texttt{xs}$ die Gültigkeit von $\texttt{rev}\ (\texttt{rev}\ (\texttt{x}\ ::\ \texttt{xs})) = \texttt{x}\ ::\ \texttt{xs}$ folgt:

$$\begin{aligned}
\texttt{rev}\ (\texttt{rev}\ (\texttt{x}\ ::\ \texttt{xs})) &= \texttt{rev}\ (\texttt{app}\ (\texttt{rev}\ \texttt{xs})\ [\texttt{x}]) \\
&\overset{a)}{=} \texttt{app}\ (\texttt{rev}\ [\texttt{x}])\ (\texttt{rev}\ (\texttt{rev}\ \texttt{xs})) \\
&\overset{IA}{=} \texttt{app}\ (\texttt{rev}\ [\texttt{x}])\ \texttt{xs} \\
&\overset{Def.\ \texttt{rev}}{=} \texttt{app}\ [\texttt{x}]\ \texttt{xs} \\
&= \texttt{x}\ ::\ \texttt{xs}
\end{aligned}$$