

# Programmierparadigmen

## Tutorium: Lazyness, Streams

Prof. Dr.-Ing. Gregor Snelting | WS 2012/2013

### LEHRSTUHL PROGRAMMIERPARADIGMEN

```

prime :: Integer -> Bool
prime n = (n>=2) && not (any (divides n) [2..n-1])
  where divides n m = n `mod` m == 0

queens :: Conf -> [Conf]
queens board =
  if (solution board) then [board]
  else flatten (map damens (filter legal (succs board) NUMTHRS; i++))

primes :: [Integer]
primes = sieve [2..]
  where sieve [] = []
        sieve (p : xs) = p : sieve [x | x <- xs, x > p]

qsort :: [Integer] -> [Integer]
qsort [] = []
qsort (p:ps) = (qsort [x | x <- ps, x <= p])
              ++ p: (qsort [x | x <- ps, x > p])

bal :: RedBlackTree t -> RedBlackTree t
bal (Node Black (Node Red (Node Red a x b) y (z) di) <- NUMTHRS; i++) (Node Red (Node Black a x b) y (Node Black thread join(callThd[i], &status);

/* After joining, print out the results and clean up.

```

$$\begin{array}{c}
 \Gamma(f) = \\
 \Gamma(f) = \forall \tau. \tau \rightarrow \text{int} \quad \Gamma \vdash f : \\
 \Gamma \vdash f : \text{int} \rightarrow \text{int} \quad \Gamma \vdash f : \forall \tau. \tau \rightarrow \text{int} \\
 \Gamma
 \end{array}$$

$$\Gamma \vdash \text{let } f = \lambda x. 2 \text{ in } f(f \text{ tr}$$

rekursive `data`-Typen: darstellbar im  $\lambda$ -Kalkül, z.B. Listen:

$$[1, 2, 3] \simeq \lambda c. \lambda n. c\ 1\ (c\ 2\ (c\ 3\ n))$$

- möglicher Typ:  $\tau_{\text{list}(\alpha)} = (\alpha \rightarrow \beta \rightarrow \beta) \rightarrow \beta \rightarrow \beta$   
 $\alpha$ : Elementtyp

- aber: Funktionstypen passen nicht

$$\text{tail} = \lambda l\ c\ n. l\ (\lambda x\ xs\ i. i\ x\ (xs\ c))\ (\lambda f. n)\ (\lambda x\ xs. xs)$$

$\vdash \text{tail} : ((\alpha_2 \rightarrow (\alpha \rightarrow \alpha_3) \rightarrow (\alpha_2 \rightarrow \alpha_3 \rightarrow \alpha_4) \rightarrow \alpha_4) \rightarrow (\alpha_5 \rightarrow \alpha_1) \rightarrow (\alpha_6 \rightarrow \alpha_7 \rightarrow \alpha_7) \rightarrow \alpha_8) \rightarrow \alpha \rightarrow \alpha_1 \rightarrow \alpha_8$

- zumindest nicht:  $\tau_{\text{list}(\alpha)} \rightarrow \tau_{\text{list}(\alpha)}$

## Erweitere $\lambda$ -Kalkül um Listen

- `Cons`, `Nil`, `head`, `tail`, `null`
- `null (Cons t1 t2)  $\Rightarrow$  False`    `null Nil  $\Rightarrow$  True`  
`head (Cons t1 t2)  $\Rightarrow$  t1`    `tail (Cons t1 t2)  $\Rightarrow$  t2`
- `head Nil  $\not\Rightarrow$  (Laufzeitfehler)`    `head 42 (Typfehler)`

Typ von `Cons`, `Nil`, `head`, `tail`, `null` abhängig von Elementtyp:

- erweiterte Typsyntax

$$\tau = \underbrace{\textit{Basistyp}}_{\text{int, bool}} \mid \underbrace{\textit{Var}}_{\alpha, \alpha_2, \beta, \gamma, \dots} \mid \underbrace{\tau_1 \rightarrow \tau_2}_{\text{2-stelliger Typkonstruktor}} \mid \underbrace{\textit{list}(\tau)}_{\text{1-stelliger Typkonstruktor}}$$

- keine fixen Typen  $\tau_{\text{Cons}}$ ,  $\tau_{\text{head}}$ , ...  $\Rightarrow$  neue *Typregeln*

## Typregeln für Listen

$$\text{CONS} \frac{}{\Gamma \vdash \text{Cons } t_1 \ t_2 : \text{list}(\tau)}$$

$$\text{NIL} \frac{}{\Gamma \vdash \text{Nil} : \text{list}(\tau)}$$

$$\text{HEAD} \frac{}{\Gamma \vdash \text{head } t : \text{list}(\tau)}$$

$$\text{TAIL} \frac{}{\Gamma \vdash \text{tail } t : \text{list}(\tau)}$$

$$\text{NULL} \frac{}{\Gamma \vdash \text{null } t : \text{bool}}$$

Typ von `Cons`, `Nil`, `head`, `tail`, `null` abhängig von Elementtyp:

- erweiterte Typsyntax

$$\tau = \underbrace{\text{Basistyp}}_{\text{int, bool}} \mid \underbrace{\text{Var}}_{\alpha, \alpha_2, \beta, \gamma, \dots} \mid \underbrace{\tau_1 \rightarrow \tau_2}_{\text{2-stelliger Typkonstruktor}} \mid \underbrace{\text{list}(\tau)}_{\text{1-stelliger Typkonstruktor}}$$

- keine fixen Typen  $\tau_{\text{Cons}}$ ,  $\tau_{\text{head}}$ , ...  $\Rightarrow$  neue *Typregeln*

## Typregeln für Listen

$$\text{CONS} \frac{\Gamma \vdash t_1 : \tau \quad \Gamma \vdash t_2 : \text{list}(\tau)}{\Gamma \vdash \text{Cons } t_1 \ t_2 : \text{list}(\tau)}$$

$$\text{NIL} \frac{}{\Gamma \vdash \text{Nil} : \text{list}(\tau)}$$

$$\text{HEAD} \frac{\Gamma \vdash t : \text{list}(\tau)}{\Gamma \vdash \text{head } t : \text{list}(\tau)}$$

$$\text{TAIL} \frac{\Gamma \vdash t : \text{list}(\tau)}{\Gamma \vdash \text{tail } t : \text{list}(\tau)}$$

$$\text{NULL} \frac{\Gamma \vdash t : \text{list}(\tau)}{\Gamma \vdash \text{null } t : \text{bool}}$$

Typ von `Cons`, `Nil`, `head`, `tail`, `null` abhängig von Elementtyp:

- erweiterte Typsyntax

$$\tau = \underbrace{\text{Basistyp}}_{\text{int, bool}} \mid \underbrace{\text{Var}}_{\alpha, \alpha_2, \beta, \gamma, \dots} \mid \underbrace{\tau_1 \rightarrow \tau_2}_{\text{2-stelliger Typkonstruktor}} \mid \underbrace{\text{list}(\tau)}_{\text{1-stelliger Typkonstruktor}}$$

- keine fixen Typen  $\tau_{\text{Cons}}$ ,  $\tau_{\text{head}}$ , ...  $\Rightarrow$  erweiterte Typannahmen

$$\begin{aligned} \Gamma_{\text{list}} = & \text{Cons} : \forall \alpha. \alpha \rightarrow \text{list}(\alpha) \rightarrow \text{list}(\alpha), \\ & \text{Nil} : \forall \alpha. \text{list}(\alpha), \\ & \text{head} : \forall \alpha. \text{list}(\alpha) \rightarrow \text{list}(\alpha), \\ & \text{tail} : \forall \alpha. \text{list}(\alpha) \rightarrow \text{list}(\alpha), \\ & \text{null} : \forall \alpha. \text{list}(\alpha) \rightarrow \text{bool} \end{aligned}$$

**Funktionen in Haskell:** polymorphe Definition erweitert  $\Gamma$  um Typschema

$$\begin{aligned} \text{length} & :: [a] \rightarrow \mathbf{Int} & \Gamma' = \Gamma, \text{length} : \forall \alpha. \text{list}(\alpha) \rightarrow \text{int} \\ \text{length list} & = \dots \end{aligned}$$

Wie zuvor:  $[t_1, t_2, \dots, t_n]$  für  $\text{Cons } t_1 (\dots (\text{Cons } t_n \text{ Nil}) \dots)$

$$\frac{\frac{\Gamma_{\text{list}} \vdash \&\& : \alpha_4 \quad \frac{\Gamma_{\text{list}} \vdash \text{null} : \alpha_6 \quad \Gamma_{\text{list}} \vdash [] : \alpha_7}{\Gamma_{\text{list}} \vdash \text{null } [] : \alpha_5}}{\Gamma_{\text{list}} \vdash \&\& (\text{null } []) : \alpha_3} \quad \frac{\Gamma_{\text{list}} \vdash \text{null} : \alpha_8 \quad \Gamma_{\text{list}} \vdash [1, 2, 3] : \alpha_9}{\Gamma_{\text{list}} \vdash \text{null } [1, 2, 3] : \alpha_4}}{\Gamma_{\text{list}} \vdash (\text{null } []) \&\& (\text{null } [1, 2, 3]) : \alpha_1}$$

$$\begin{aligned} C = \{ & \alpha_9 = \text{list}(\text{int}), \alpha_3 = \alpha_4 \rightarrow \alpha_1, \alpha_4 = \alpha_5 \rightarrow \alpha_3, \\ & \alpha_4 = \text{bool} \rightarrow (\text{bool} \rightarrow \text{bool}), \alpha_6 = \alpha_7 \rightarrow \alpha_5, \alpha_8 = \alpha_9 \rightarrow \alpha_4 \\ & \alpha_6 = \text{????} \\ & \alpha_8 = \text{????} \\ & \alpha_7 = \text{????} \} \end{aligned}$$

$$\begin{aligned} \Gamma_{\text{list}} = \dots \\ & \text{null} : \forall \alpha. \text{list}(\alpha) \rightarrow \text{bool}, \\ & \text{Nil} : \forall \alpha. \text{list}(\alpha) \\ & \dots \end{aligned}$$

Wie zuvor:  $[t_1, t_2, \dots, t_n]$  für  $\text{Cons } t_1 (\dots (\text{Cons } t_n \text{ Nil}) \dots)$

$$\frac{\frac{\Gamma_{\text{list}} \vdash \&\& : \alpha_4 \quad \frac{\Gamma_{\text{list}} \vdash \text{null} : \alpha_6 \quad \Gamma_{\text{list}} \vdash [] : \alpha_7}{\Gamma_{\text{list}} \vdash \text{null } [] : \alpha_5}}{\Gamma_{\text{list}} \vdash \&\& (\text{null } []) : \alpha_3} \quad \frac{\Gamma_{\text{list}} \vdash \text{null} : \alpha_8 \quad \Gamma_{\text{list}} \vdash [1, 2, 3] : \alpha_9}{\Gamma_{\text{list}} \vdash \text{null } [1, 2, 3] : \alpha_4}}{\Gamma_{\text{list}} \vdash (\text{null } []) \&\& (\text{null } [1, 2, 3]) : \alpha_1}$$

$$\begin{aligned} C = \{ & \alpha_9 = \text{list}(\text{int}), \alpha_3 = \alpha_4 \rightarrow \alpha_1, \alpha_4 = \alpha_5 \rightarrow \alpha_3, \\ & \alpha_4 = \text{bool} \rightarrow (\text{bool} \rightarrow \text{bool}), \alpha_6 = \alpha_7 \rightarrow \alpha_5, \alpha_8 = \alpha_9 \rightarrow \alpha_4 \\ & \alpha_6 = \text{list}(\beta_1) \rightarrow \text{bool} \\ & \alpha_8 = \text{list}(\beta_2) \rightarrow \text{bool} \\ & \alpha_7 = \text{list}(\beta_3) \} \end{aligned}$$

$$\begin{aligned} \Gamma_{\text{list}} = \dots \\ \text{null} : \forall \alpha. \text{list}(\alpha) \rightarrow \text{bool}, \\ \text{Nil} : \forall \alpha. \text{list}(\alpha) \\ \dots \end{aligned}$$