University of Innsbruck — **Institute of Computer Science**

2nd Exam — **April 16, 2010**

Functional Programming | **WS 2009/2010** | **LVA 703017**

Solutions

1. Consider the λ-term $t = (\lambda xyz.x\ z\ (y\ z))\ (\lambda xy.x)\ (\lambda x.x)\ (\lambda x.x)$.

[12] (a) Reduce t stepwise to normal form, using the leftmost innermost strategy.

Solution.

$$
\begin{aligned}
&\underline{(\lambda xyz.x\ z\ (y\ z))\ (\lambda xy.x)}\ (\lambda x.x)\ (\lambda x.x)\\
&\to_\beta (\lambda yz.\underline{(\lambda xy.x)\ z}\ (y\ z))\ (\lambda x.x)\ (\lambda x.x)\\
&\to_\beta (\lambda yz.\underline{(\lambda y.z)\ (y\ z)})\ (\lambda x.x)\ (\lambda x.x)\\
&\to_\beta \underline{(\lambda yz.z)\ (\lambda x.x)}\ (\lambda x.x)\\
&\to_\beta \underline{(\lambda z.z)\ (\lambda x.x)}\\
&\to_\beta \lambda x.x
\end{aligned}
$$

[13] (b) Reduce t stepwise to normal form, using the leftmost outermost strategy.

Solution.

$$
\begin{aligned}
&\underline{(\lambda xyz.x\ z\ (y\ z))\ (\lambda xy.x)}\ (\lambda x.x)\ (\lambda x.x)\\
&\to_\beta \underline{(\lambda yz.(\lambda xy.x)\ z\ (y\ z))\ (\lambda x.x)}\ (\lambda x.x)\\
&\to_\beta \underline{(\lambda z.(\lambda xy.x)\ z\ ((\lambda x.x)\ z))\ (\lambda x.x)}\\
&\to_\beta \underline{(\lambda xy.x)\ (\lambda x.x)}\ ((\lambda x.x)\ (\lambda x.x))\\
&\to_\beta \underline{(\lambda yx.x)\ ((\lambda x.x)\ (\lambda x.x))}\\
&\to_\beta \lambda x.x
\end{aligned}
$$

2. Consider the OCaml type `type tree = E | N of tree * tree` together with the function

```
let rec mirror = function
  | E         -> E
  | N (l, r) -> N (mirror r, mirror l)
```

Prove by induction that `mirror` (`mirror` t) $= t$ for every value t of type `tree`.

[5] (a) Base case.

Solution.

Base Case ($t =$ `E`). By applying the definitions of `mirror`, we prove the base case as follows:
`mirror` (`mirror E`) $=$ `mirror E` $=$ `E`.

[20] (b) Step case.

Solution.

Step Case ($t =$ `N` (l, r)). The IHs are `mirror` (`mirror` l) $= l$ and `mirror` (`mirror` r) $= r$.

$$
\begin{aligned}
\texttt{mirror}\ (\texttt{mirror}\ (\texttt{N}\ (l, r))) &= \texttt{mirror}\ (\texttt{N}\ (\texttt{mirror}\ r, \texttt{mirror}\ l))\\
&= \texttt{N}\ (\texttt{mirror}\ (\texttt{mirror}\ l), \texttt{mirror}\ (\texttt{mirror}\ r))\\
&= \texttt{N}\ (l, r) && \text{by IHs}
\end{aligned}
$$

3. Consider the OCaml function

```
let rec repeat i n =
  if n < 1 then []
          else i :: repeat i (n - 1)
```

[12] (a) Implement a tail-recursive variant of `repeat`.

Solution.

```
let repeat' i n =
  let rec rep acc n =
    if n < 1 then acc
             else rep (i :: acc) (n - 1)
  in
  rep [] n
```

[13] (b) Use tupling to implement a function `fraction : 'a -> 'a list -> float` that determines for a given element x in a list xs, the ratio (between 0 and 1) it constitutes to the whole list, e.g.,

`fraction 'a' ['a'; 'b'; 'c'; 'a']` $=$ `0.5`

Solution.

```
let fraction x ys =
  let rec length_count (len, num) = function
    | []       -> (len, num)
    | y :: ys -> if x = y
      then length_count (len + 1, num + 1) ys
      else length_count (len + 1, num)     ys
  in
  let (len, num) = length_count (0, 0) ys in
  if len = 0 then 0.0 else float_of_int num /. float_of_int len
```

4. Consider the environment $E = \{1 : \mathsf{int}, 2 : \mathsf{int}, :: : \mathsf{int} \to \tau \to \tau, \mathsf{hd} : \tau \to \mathsf{int}, [] : \tau, \mathsf{tl} : \tau \to \tau\}$, where we use the abbreviation $\tau = \mathsf{list}(\mathsf{int})$ and :: is assumed to be a right-associative infix operator.

[12] (a) Prove the typing judgment $E \vdash \mathbf{let}\ x = \mathsf{tl}\ (1 :: 2 :: [])\ \mathbf{in}\ \mathsf{hd}\ x : \mathsf{int}$.

Solutions

Solution.

$$\frac{}{E \vdash \mathsf{tl} : \tau \to \tau}\,(\mathsf{ref})$$

$$\frac{\dfrac{}{E \vdash :: : \mathsf{int} \to \tau \to \tau}\,(\mathsf{ref}) \qquad \dfrac{}{E \vdash 1 : \mathsf{int}}\,(\mathsf{ref})}{E \vdash (::)\; 1 : \tau \to \tau}\,(\mathsf{app})$$

$$\frac{\dfrac{}{E \vdash :: : \mathsf{int} \to \tau \to \tau}\,(\mathsf{ref}) \qquad \dfrac{}{E \vdash 2 : \mathsf{int}}\,(\mathsf{ref})}{E \vdash (::)\; 2 : \tau \to \tau}\,(\mathsf{app})$$

$$\frac{E \vdash (::)\; 2 : \tau \to \tau \qquad \dfrac{}{E \vdash [\,] : \tau}\,(\mathsf{ref})}{E \vdash 2 :: [\,] : \tau}\,(\mathsf{app})$$

$$\frac{E \vdash (::)\; 1 : \tau \to \tau \qquad E \vdash 2 :: [\,] : \tau}{E \vdash 1 :: 2 :: [\,] : \tau}\,(\mathsf{app})$$

$$\frac{E \vdash \mathsf{tl} : \tau \to \tau \qquad E \vdash 1 :: 2 :: [\,] : \tau}{E \vdash \mathsf{tl}\,(1 :: 2 :: [\,]) : \tau}\,(\mathsf{app})$$

$$\frac{\dfrac{}{E' \vdash \mathsf{hd} : \tau \to \mathsf{int}}\,(\mathsf{ref}) \qquad \dfrac{}{E' \vdash x : \tau}\,(\mathsf{ref})}{E' = E, x : \tau \vdash \mathsf{hd}\; x : \mathsf{int}}\,(\mathsf{app})$$

$$\frac{E \vdash \mathsf{tl}\,(1 :: 2 :: [\,]) : \tau \qquad E' = E, x : \tau \vdash \mathsf{hd}\; x : \mathsf{int}}{E \vdash \mathbf{let}\; x = \mathsf{tl}\,(1 :: 2 :: [\,])\; \mathbf{in}\; \mathsf{hd}\; x : \mathsf{int}}\,(\mathsf{let})$$

[13] (b) Solve the unification problem:

$$\alpha_3 \to \mathsf{list}(\alpha_3) \to \mathsf{list}(\alpha_3) \approx \alpha_2 \to \alpha_1 \to \alpha_4;$$
$$\mathsf{bool} \approx \alpha_2;$$
$$\mathsf{list}(\alpha_0) \approx \alpha_1;$$
$$\mathsf{list}(\alpha_0) \approx \alpha_4$$

Solution.

$$\alpha_3 \to \mathsf{list}(\alpha_3) \to \mathsf{list}(\alpha_3) \approx \alpha_2 \to \alpha_1 \to \alpha_4;$$
$$\mathsf{bool} \approx \alpha_2; \mathsf{list}(\alpha_0) \approx \alpha_1; \mathsf{list}(\alpha_0) \approx \alpha_4$$
$$\Rightarrow_{\iota}^{(\mathsf{d}_2)+}$$
$$\alpha_3 \approx \alpha_2; \mathsf{list}(\alpha_3) \approx \alpha_1; \mathsf{list}(\alpha_3) \approx \alpha_4; \mathsf{bool} \approx \alpha_2; \mathsf{list}(\alpha_0) \approx \alpha_1; \mathsf{list}(\alpha_0) \approx \alpha_4$$
$$\Rightarrow_{\{\alpha_3/\alpha_2\}}^{(\mathsf{v}_1)}$$
$$\mathsf{list}(\alpha_2) \approx \alpha_1; \mathsf{list}(\alpha_2) \approx \alpha_4; \mathsf{bool} \approx \alpha_2; \mathsf{list}(\alpha_0) \approx \alpha_1; \mathsf{list}(\alpha_0) \approx \alpha_4$$
$$\Rightarrow_{\{\alpha_1/\mathsf{list}(\alpha_2)\}}^{(\mathsf{v}_2)}$$
$$\mathsf{list}(\alpha_2) \approx \alpha_4; \mathsf{bool} \approx \alpha_2; \mathsf{list}(\alpha_0) \approx \mathsf{list}(\alpha_2); \mathsf{list}(\alpha_0) \approx \alpha_4$$
$$\Rightarrow_{\{\alpha_4/\mathsf{list}(\alpha_2)\}}^{(\mathsf{v}_2)}$$
$$\mathsf{bool} \approx \alpha_2; \mathsf{list}(\alpha_0) \approx \mathsf{list}(\alpha_2); \mathsf{list}(\alpha_0) \approx \mathsf{list}(\alpha_2)$$
$$\Rightarrow_{\{\alpha_2/\mathsf{bool}\}}^{(\mathsf{v}_2)}$$
$$\mathsf{list}(\alpha_0) \approx \mathsf{list}(\mathsf{bool}); \mathsf{list}(\alpha_0) \approx \mathsf{list}(\mathsf{bool})$$
$$\Rightarrow_{\iota}^{(\mathsf{d}_1)+}$$
$$\alpha_0 \approx \mathsf{bool}; \alpha_0 \approx \mathsf{bool};$$
$$\Rightarrow_{\{\alpha_0/\mathsf{bool}\}}^{(\mathsf{v}_1)}$$
$$\mathsf{bool} \approx \mathsf{bool};$$
$$\Rightarrow_{\iota}^{(\mathsf{t})}$$
$$\square$$

The resulting substitution is

$$\{\alpha_0/\mathsf{bool}, \alpha_1/\mathsf{list}(\mathsf{bool}), \alpha_2/\mathsf{bool}, \alpha_3/\mathsf{bool}, \alpha_4/\mathsf{list}(\mathsf{bool})\}$$