Friedrich-Alexander-Universität
Technische Fakultät

Theorie der Programmierung

Übung 10 - System F

Leon Vatthauer

3. Juli 2023

System F

System F ist eine Erweiterung des einfach getypten λ-Kalküls um Polymorphie. Wir erweitern die Typgrammatik:

$$\alpha, \beta ::= a \mid \alpha \to \beta \mid \forall a.\alpha \qquad (a \in \mathbf{V})$$

Die Typisierungsregeln sind:

$$(\mathrm{Ax})\ \frac{}{\Gamma \vdash x : \alpha}\ (x : \alpha \in \Gamma) \quad (\to_i)\ \frac{\Gamma[x \mapsto \alpha] \vdash t : \beta}{\Gamma \vdash \lambda x.t : \alpha \to \beta} \quad (\forall_i)\ \frac{\Gamma \vdash s : \alpha}{\Gamma \vdash s : \forall a.\alpha}\ a \notin FV(\Gamma)$$

$$(\to_e)\ \frac{\Gamma \vdash t : \alpha \to \beta \qquad \Gamma \vdash s : \alpha}{\Gamma \vdash t\ s : \beta} \qquad (\forall_e)\ \frac{\Gamma \vdash s : \forall a.\alpha}{\Gamma \vdash s : (\alpha[\beta/a])}$$

Aufgabe 1

Produkte in System F (à la Curry)

Wir kodieren das kartesische Produkt der Typen a und b in System F unter Verwendung des Typs
$(a \times b) := \forall r.(a \to b \to r) \to r$

1. Für diese Kodierung ist die "Konstruktions-Funktion" *pair*, welche aus zwei Elementen der Typen a bzw. b ein Paar des Typs $(a \times b)$ konstruiert, wie folgt definiert:

   $$pair = \lambda x\ y.(\lambda f.\ f\ x\ y)$$

   Zeigen Sie, dass $\vdash pair : \forall a\ b.\ a \to b \to (a \times b)$ in System F gilt.

Aufgabe 1

Produkte in System F (à la Curry)

Wir kodieren das kartesische Produkt der Typen a und b in System F unter Verwendung des Typs $(a \times b) := \forall r.(a \to b \to r) \to r$

1. Für diese Kodierung ist die “Konstruktions-Funktion” *pair*, welche aus zwei Elementen der Typen a bzw. b ein Paar des Typs $(a \times b)$ konstruiert, wie folgt definiert:

   $$pair = \lambda x\ y.(\lambda f.\ f\ x\ y)$$

   Zeigen Sie, dass $\vdash pair : \forall a\ b.\ a \to b \to (a \times b)$ in System F gilt.

2. Geben Sie zu jeder der folgenden Funktionssignaturen eine Implementierung der jeweiligen Funktion (d.h. einen λ-Term) an und zeigen Sie, dass Ihre Implementierung den benötigten Typ hat:
   (a) $fst : \forall a\ b.\ (a \times b) \to a$
   (b) $snd : \forall a\ b.\ (a \times b) \to b$

Aufgabe 1

Produkte in System F (à la Curry)

Wir kodieren das kartesische Produkt der Typen a und b in System F unter Verwendung des Typs $(a \times b) := \forall r.(a \to b \to r) \to r$

3. Schreiben Sie unter Verwendung der obigen Funktionen eine weitere Funktion

$$swap : \forall a\, b.(a \times b) \to (b \times a)$$

   und zeigen Sie, dass sie den korrekten Typ hat. Finden Sie also einen λ-Term s, so dass $\Gamma \vdash s : \forall a\, b.(a \times b) \to (b \times a)$, wobei

$$\begin{aligned}\Gamma := \{&pair : \forall a\, b.a \to b \to (a \times b),\\ &fst : \forall a\, b.\ (a \times b) \to a,\\ &snd : \forall a\, b.\ (a \times b) \to b\}\end{aligned}$$

Aufgabe 2

Listen in System F (à la Curry)

Listen können in System F unter Verwendung des folgenden Typs kodiert werden:

$$\mathbf{List}\ a\ :=\ \forall r.\ r \to (a \to r \to r) \to r$$

In diesem Fall ergeben sich die folgenden "Konstruktor-Funktionen":

$$\begin{aligned} \mathtt{nil} &= \lambda\ \mathtt{u}\ \mathtt{f}.\ \mathtt{u} \\ \mathtt{cons} &= \lambda \mathtt{x}\ \mathtt{l}.\lambda\ \mathtt{u}\ \mathtt{f}.\ \mathtt{f}\ \mathtt{x}\ (\mathtt{l}\ \mathtt{u}\ \mathtt{f}) \end{aligned}$$

Für einen gegebenen (und durch `nil` und `cons` konstruierten) Term `t` des Typs **List** a verhält sich der Term '$\mathtt{t}\ \mathtt{u}\ (\lambda \mathtt{x}\ \mathtt{l}.\ \mathtt{s})$' also genau so wie eine Funktion `f` für die für alle x (des Typs a) und alle l (des Typs **List** a) gilt:

$$\begin{aligned} f\ nil &\to^*_{\beta\delta} u \\ f\ (cons\ x\ nil) &\to^*_{\beta\delta} s[l \mapsto f\ l] \end{aligned}$$

1. Zeigen Sie, dass:
   (a) $\vdash \mathtt{nil} : \forall\ a.\ \mathbf{List}\ a$
   (b) $\vdash \mathtt{cons} : \forall\ a.a \to \mathbf{List}\ a \to \mathbf{List}\ a$

Aufgabe 2

Listen in System F (à la Curry)

Listen können in System F unter Verwendung des folgenden Typs kodiert werden:

$$\texttt{\textbf{List} a := } \forall r.\, r \to (a \to r \to r) \to r$$

In diesem Fall ergeben sich die folgenden “Konstruktor-Funktionen”:

```
nil  = λ u f. u
cons = λx l.λ u f. f x (l u f)
```

2. Schreiben Sie eine Funktion `length`, welche die Länge einer Liste berechnet. Es soll gelten:

$$\begin{aligned} \texttt{length nil} &\to^*_{\beta\delta} \texttt{zero} \\ \texttt{length (cons x l)} &\to^*_{\beta\delta} \texttt{succ (length l)} \end{aligned}$$

Zeigen Sie, dass $\Gamma_0 \vdash \texttt{length : } \forall\ \texttt{a. }\texttt{\textbf{List} a} \to \mathbb{N}$, wobei $\Gamma_0 := \{\texttt{zero} : \mathbb{N}, \mathit{succ} : \mathbb{N} \to \mathbb{N}\}$.