Friedrich-Alexander-Universität
Technische Fakultät

FAU

Theorie der Programmierung

Übung 11 — System F

Leon Vatthauer

18. Juli 2025

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:

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

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

Aufgabe 1

Church-Kodierung im System F

Die Church-Kodierung eines induktiven Datentyps im System F entspricht dem Typ seiner Fold-Funktion bis auf Weglassen des letzten Arguments.

1. Der Typ der booleschen Wahrheitswerte (mit Fold-Funktion $\mathit{foldB} : r \to r \to \mathbf{Bool} \to r$) wird als $\mathbb{B} := \forall r.r \to r \to r$ kodiert. Zeigen Sie, dass $\vdash \mathit{true} : \mathbb{B}$ und $\vdash \mathit{false} : \mathbb{B}$, wobei $\mathit{true} = \lambda x\ y.x$ und $\mathit{false} = \lambda x\ y.y$.

Aufgabe 1

Church-Kodierung im System F

Die Church-Kodierung eines induktiven Datentyps im System F entspricht dem Typ seiner Fold-Funktion bis auf Weglassen des letzten Arguments.

1. Der Typ der booleschen Wahrheitswerte (mit Fold-Funktion $\mathit{foldB} : r \to r \to \mathbf{Bool} \to r$) wird als $\mathbb{B} := \forall r.r \to r \to r$ kodiert. Zeigen Sie, dass $\vdash \mathit{true} : \mathbb{B}$ und $\vdash \mathit{false} : \mathbb{B}$, wobei $\mathit{true} = \lambda x\ y.x$ und $\mathit{false} = \lambda x\ y.y$.
2. Der Typ **Nat** der natürlichen Zahlen (mit Fold-Funktion $\mathit{foldN} : r \to (r \to r) \to \mathbf{Nat} \to r$) wird als $\mathbb{N} := \forall r.(r \to r) \to r \to r$ kodiert. Zeigen Sie, dass $\vdash \mathit{succ} : \mathbb{N} \to \mathbb{N}$, wobei $\mathit{succ} = \lambda n.\lambda f\ a.f\ (n\ f\ a)$.

Aufgabe 1

Church-Kodierung im System F

Die Church-Kodierung eines induktiven Datentyps im System F entspricht dem Typ seiner Fold-Funktion bis auf Weglassen des letzten Arguments.

1. Der Typ der booleschen Wahrheitswerte (mit Fold-Funktion $\mathit{foldB} : r \to r \to \mathbf{Bool} \to r$) wird als $\mathbb{B} := \forall r.r \to r \to r$ kodiert. Zeigen Sie, dass $\vdash \mathit{true} : \mathbb{B}$ und $\vdash \mathit{false} : \mathbb{B}$, wobei $\mathit{true} = \lambda x\ y.x$ und $\mathit{false} = \lambda x\ y.y$.
2. Der Typ **Nat** der natürlichen Zahlen (mit Fold-Funktion $\mathit{foldN} : r \to (r \to r) \to \mathbf{Nat} \to r$) wird als $\mathbb{N} := \forall r.(r \to r) \to r \to r$ kodiert. Zeigen Sie, dass $\vdash \mathit{succ} : \mathbb{N} \to \mathbb{N}$, wobei $\mathit{succ} = \lambda n.\lambda f\ a.f\ (n\ f\ a)$.
3. Zeigen Sie außerdem, dass $\Gamma \vdash \mathit{add} : \mathbb{N} \to \mathbb{N} \to \mathbb{N}$, wobei $\Gamma = \mathit{zero} : \mathbb{N}, \mathit{succ} : \mathbb{N} \to \mathbb{N}$ und $\mathit{add} = \lambda n\ m.n\ \mathit{succ}\ m$.

Aufgabe 2

Listen in System F (à la Curry)

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

$$\mathbb{L}\ a := \forall r.r \to (a \to r \to r) \to r$$

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

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

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

$$\begin{aligned} f\ nil &\to_\beta u \\ f\ (cons\ x\ l) &\to_\beta s\ [l \mapsto f\ l] \end{aligned}$$

1. Es sei $l_0 = cons\ a\ (cons\ b\ nil)$ eine zweielementige Liste. Berechnen Sie die β-Normalform von $l_0\ u\ f$, wobei f und u beliebige Variablen sind.

Aufgabe 2

Listen in System F (à la Curry)

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

$$\mathbb{L}\ a := \forall r.r \to (a \to r \to r) \to r$$

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

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

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

$$\begin{aligned} f\ nil &\to_\beta u \\ f\ (cons\ x\ l) &\to_\beta s\ [l \mapsto f\ l] \end{aligned}$$

2. Zeigen Sie, dass:
   (a) $\vdash$ `nil : ∀ a. 𝕃 a`
   (b) $\vdash$ `cons : ∀ a.a → 𝕃 a → 𝕃 a`

Aufgabe 2

Listen in System F (à la Curry)

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

$$\mathbb{L}\ a := \forall r.r \to (a \to r \to r) \to r$$

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

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

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

$$\begin{aligned} f\ nil &\to_\beta u \\ f\ (cons\ x\ l) &\to_\beta s\ [l \mapsto f\ l] \end{aligned}$$

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

$$\begin{aligned} \texttt{length nil} &\to_\beta \texttt{zero} \\ \texttt{length (cons x l)} &\to_\beta \texttt{succ (length l)} \end{aligned}$$

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