Institut für Informatik
der Universität München
Dr. Andreas Abel

SS 2007

1. Juni 2007

Übungen zur Vorlesung
Typsysteme

Blatt 7

Aufgabe P-20 (λ-Terme als $\mathsf{F}_{<:}$-Typen): Betrachten Sie die Kodierung von λ-Termen und Stacks in das Typsystem von $\mathsf{F}_{<:}$ aus der Vorlesung. Geben Sie eine Typherleitung für $((\lambda x x)\,(\lambda x x)\star,\ [\,]) \longrightarrow^* (\star,\ [\,])$.

Aufgabe P-21 (Funktionale Objektkodierung): Wir kodieren eine Klasse von Zählern, die ausgelesen und inkrementiert werden können. Der Zustand wird durch den Verbundtyp

$$\mathsf{CounterR} = \{\mathsf{value} : \mathsf{int}\}$$

repräsentiert; bei der Konstruktion eines Zählers wird value auf 0. Die Schnittstelle (interface) ist der Verbundtyp

$$\mathsf{CounterM} = \lambda R.\,\{\mathsf{get} : R \to \mathsf{int},\ \mathsf{inc} : R \to R\}$$

wobei R für den versteckten internen Zustand des Objekts steht. Das Vestecken geschieht mittels existentieller Typen, so konstruiert

$$\mathsf{Object} = \lambda M : * \to *.\exists R.\,\{\mathsf{fields} : R,\ \mathsf{methods} : M\,R\}$$

den Typ der Objekte mit Schnittstelle M. Z.B.

$$\mathsf{Counter} = \mathsf{Object}\,\mathsf{CounterM}.$$

Implementieren Sie folgende Funktionen

$$\begin{array}{lcl}
\mathsf{makeObject} & : & \forall M : * \to *.\,\forall R.\,R \to M\,R \to \mathsf{Object}\,M \\
\mathsf{counterClass} & : & \mathsf{CounterM}\ \mathsf{CounterR} \\
\mathsf{makeCounter} & : & \mathsf{Counter} \\
\mathsf{makeCounter} & = & \mathsf{makeObject}\ [\mathsf{CounterM}]\ [\mathsf{CounterR}]\ \{\mathsf{value} = 0\}\ \mathsf{counterClass}
\end{array}$$

Aufgabe H-21 (Bounded Existentials): Das System $\mathsf{F}_{<:}$ kann um be-

schränkte existentielle Typen wie folgt erweitert werden.

$$\frac{\Gamma, \exists X <: A \vdash B \ \mathsf{ok}}{\Gamma \vdash \exists X <: A.\, B \ \mathsf{ok}} \qquad \frac{\Gamma, X <: A \vdash B <: C}{\Gamma \vdash \exists X <: A.\, B <: \exists X <: A.\, C}$$

$$\frac{\Gamma \vdash t : B[A'/X] \qquad \Gamma \vdash A' <: A}{\Gamma \vdash (\mathsf{pack}\ A', t\ \mathsf{as}\ \exists X <: A.\, B) : \exists X <: A.\, B}$$

$$\frac{\Gamma \vdash r : \exists X <: A.\, B \qquad \Gamma, X <: A, x : B \vdash s : C}{\Gamma \vdash (\mathsf{let}\ X <: A, x = r\ \mathsf{in}\ s\ \mathsf{return}\ C) : C}$$

$$\mathsf{let}\ X <: A, x = (\mathsf{pack}\ A', v\ \mathsf{as}\ \exists X <: A.\, B)\ \mathsf{in}\ s\ \mathsf{return}\ C \longrightarrow s[A'/X][v/x]$$

Jedoch lassen sich diese existentiellen Typen bereits in $\mathsf{F}_{<:}$ *kodieren*: $\exists X <: A.\, B = (\forall Y <: \mathsf{Top}.\, (\forall X <: A.\, B \to Y) \to Y$. Definieren Sie noch pack und let und zeigen Sie, dass obige 5 Regeln herleitbar sind (dabei muss $\longrightarrow$ jedoch durch $\longrightarrow^+$ ersetzt werden).

Aufgabe H-22 (Erweiterte Klassen): Erweitern Sie die obige Zählerklasse um eine Resetfunktion reset : $R \to R$ und definieren Sie ResetCounterM, ResetCounterM, resetCounterClass und makeResetCounter entsprechend. Beweisen Sie, dass ResetCounter <: Counter.

Aufgabe H-23 (Objektkodierung: Senden von Nachrichten): Definieren Sie Funktionen

$$\begin{array}{lcl} \mathsf{sendGet} & : & \forall C <: \mathsf{Counter}.\, C \to \mathsf{int} \\ \mathsf{sendInc} & : & \forall C <: \mathsf{Counter}.\, C \to C \\ \mathsf{sendReset} & : & \forall C <: \mathsf{ResetCounter}.\, C \to C \end{array}$$

und werten Sie folgendes Programm aus ($C = \mathsf{ResetCounter}$).

$$\mathsf{sendGet}[C]\,(\mathsf{sendReset}[C]\,(\mathsf{sendInc}[C]\,\mathsf{makeResetCounter}))$$

Abgabe der Hausaufgaben H-X zum Beginn der nächsten Übungsstunde.