P. Thiemann, M. Neubauer Wintersemester 2004/2005

Compilerbau

http://proglang.informatik.uni-freiburg.de/teaching/compilerbau/2004/

Übungsblatt 10

Abgabe: 26.01.2005

Aufgabe 1 (Tail-Call-Elimination):

Verbessere die Transformation $\mathcal{R}$ aus der Vorlesung so, dass keine unnötigen Continuations mehr erzeugt werden und Funktionsaufrufe ohne Kontext immer zu Tail-Calls werden.

Aufgabe 2 (Zwischensprache mit expliziten Rücksprungadressen):

Die folgende Variante der sequentiellen Zwischensprache benennt nicht nur Zwischenergebnisse explizit sondern gibt zu jeder Funktions immer auch eine Fortsetzungsfunktion (d.h. die "Rücksprungadresse") explizit mit an:

Definitions

$$d ::= f = \lambda^{@} z.\lambda x.\lambda^{\#} k.s$$

Trivial Expressions

$$\begin{array}{rcl} t & ::= & x \mid c \\ & \mid & b\,\overline{t} \\ & \mid & f@z \\ & \mid & \langle x_1, \ldots, x_n \rangle \mid t \downarrow i \end{array}$$

Serious Expressions

$$\begin{array}{rcl} s & ::= & t\,x\,k \\ & \mid & \texttt{let}\ x = t\ \texttt{in}\ s \\ & \mid & \texttt{let}\ x = b\,\overline{x}\ \texttt{in}\ s \\ & \mid & \texttt{if}\ x\ s\ s \\ & \mid & \texttt{letcont}\ k@\langle x_1 \ldots x_n \rangle = \lambda x.s\ \texttt{in}\ s \\ & \mid & \texttt{yield}\ k\ x \end{array}$$

Funktionsdefinitionen bekommen einen zusätzlichen, speziellen Parameter k, der mögliche Fortsetzungsfunktionen repräsentiert. Funktionen werden nun verlassen, indem die Kontrolle explizit an eine bestimmte Fortsetzung mit Hilfe von `yield` weitergegeben wird.

Gib neue, an die neue Zwischensprache angepasste Varianten der drei Transformationsschritte aus der Vorlesung an!

Aufgabe 3 (Inlining):

Passe ebenso die Inlining-Transformation aus der Vorlesung an die alternative Zwischensprache aus Aufgabe 2 an.