

Theorie der Programmierung (WS 2003/04)

Übungsblatt 5

Aufgabe 1

Denken Sie sich abgeleitete small step Regeln für die n -stellige Abstraktion und das n -stellige **let val** aus und leiten Sie die Regeln für $n = 2$ her. (Da uns die genaue Anzahl der small steps nicht interessiert, darf man hier mehrere small steps zu einem zusammenfassen.)

Aufgabe 2

Wir hätten die Kernsyntax unserer Programmiersprache noch weiter verkleinern können, wenn wir anstelle von bedingten Ausdrücken einen Operator *cond* aufgenommen hätten so, dass sich **if** e_0 **then** e_1 **else** e_2 als syntaktischer Zucker für die Applikation $cond(e_0, \lambda id : \mathbf{unit}. e_1, \lambda id : \mathbf{unit}. e_2)$ mit $id \notin free(e_1) \cup free(e_2)$ auffassen lässt. (Warum kann $cond(e_0, e_1, e_2)$ nicht funktionieren?)

Wie müssen die Typregel und die small step Regeln für *cond* aussehen, damit sich aus ihnen die Typregel und die small step Regeln für **if** e_0 **then** e_1 **else** e_2 ableiten lassen? Führen Sie diese Ableitungen durch.

Kann man auch **let val** $id = e_1$ **in** e_2 **end** als syntaktischen Zucker für eine geeignete Applikation auffassen?

Aufgabe 3

Unsere small step Semantik benutzt das *call by value* Prinzip bei der Parameterübergabe, d.h. der aktuelle Parameter wird ausgewertet, bevor er für den formalen Parameter eingesetzt wird. Im Gegensatz dazu bedeutet *call by name*, dass der aktuelle Parameter unausgewertet eingesetzt wird.

Ändern Sie unsere small step Regeln so ab, dass Sie eine *call by name* Semantik erhalten. Achten Sie darauf, dass Ihre Semantik die üblichen ‘guten Eigenschaften’ (Determinismus, Preservation, Progress) besitzt. Geben Sie einen Ausdruck an, der unterschiedliche Resultate in den beiden Semantiken liefert.

Aufgabe 4

Führen Sie den Beweis von Satz 5 (“Progress”) zu Ende.

Aufgabe 5

Bestimmen Sie die big step Semantik des folgenden Programms

```
let val twice =  $\lambda f : \mathbf{int} \rightarrow \mathbf{int}. \lambda x : \mathbf{int}. f (f x)$   
  val square =  $\lambda x : \mathbf{int}. x * x$   
in twice square 5  
end
```