Dr. Annette Bieniusa
Mathias Weber, M. Sc.
Peter Zeller, M. Sc.

TU Kaiserslautern
Fachbereich Informatik
AG Softwaretechnik

Übungsblatt 14p: Software-Entwicklung 1 (WS 2017/18)

Ausgabe: 06.02.18
Abgabe: Bearbeitung in Übung

Aufgabe 1 Präsenzaufgabe zu Lambdas

a) Verwenden Sie die formale Definition für freie Variablen und berechnen Sie damit die Menge der freien Variablen in folgendem Lambda-Term. Geben Sie dabei alle Zwischenschritte an.

$\Big((\lambda x \to \lambda f \to f\ a\ x)\ x\Big)$

b) Sind die folgenden Substitutionen erlaubt? Falls nicht, begründen Sie Ihre Antwort. Falls ja, berechnen Sie für die Substitution das Ergebnis mit Hilfe der formalen Definition. Geben Sie dabei alle Zwischenschritte an.

1. $\Big(h\ x\ (\lambda x \to f\ x)\Big)[42/x]$
2. $\Big(h\ x\ (\lambda x \to f\ x)\Big)[g\ x/f]$

c) Bringen Sie die folgenden Lambda-Terme in eine β-Normalform. Geben Sie dabei alle Zwischenschritte an (die erforderlichen Substitution sollen dabei für β-Konversionen explizit angegeben werden, dürfen dann aber in einem Schritt ausgewertet werden; α-Konversionen dürfen in einem Schritt durchgeführt werden).

Die Lambda-Terme in dieser Aufgabe enthalten Zahlen und mathematische Operationen, welche mathematisch auszuwerten sind.

1. $(\lambda x \to x \cdot x)\ 3$
2. $\Big(\lambda x\ y \to (\lambda x\ y \to a\ y\ x)\ y\ x\Big)$

Definitionen

Freie Variablen Die Menge der freien Variablen eines Lambda-Ausdrucks `M`, bezeichnet als `FV(M)`, ist rekursiv über die Struktur des Ausdrucks definiert:

- `FV(x) = {x}` für eine Variable x
- `FV(λx → e) = FV(e) \ {x}`
- `FV(f g) = FV(f) ∪ FV(g)`

Substitution

- `x[t/x] = t`
- `y[t/x] = y` falls $x \neq y$
- `(f g)[t/x] = (f[t/x] g[t/x])`
- `(λx → e)[t/x] = (λx → e)`
- `(λy → e)[t/x] = (λy → e[t/x])` falls $x \neq y$ und $y \notin FV(t)$

Semantik von Lambda-Ausdrücken Die Bedeutung (Semantik) von Lambda-Ausdrücken lässt sich mit Hilfe der Substitution und zwei Umformungsregeln erklären. Die Umformungsregeln sind dabei die α-Konversion und die β-Konversion.

Die **α-Konversion** erlaubt das Umbenennen von gebundenen Variablen. Der Term `λx → t` lässt sich äquivalent umformen zum Term `λy → (t[y/x])`, wenn y nicht bereits als freie Variable im Term `t` vorkommt und die Substitution erlaubt ist.

Beispiel: `λa → f (f a)` ist äquivalent zu `λb → f (f b)`, aber nicht zu `λf → f (f f)`

Die **β-Konversion** erklärt die Auswertung von Funktionsaufrufen. Wenn der linke Term eines Funktionsaufrufs eine Abstraktion ist, so kann der Aufruf ausgewertet werden, indem im Funktionsrumpf der Parameter durch den aktuell gegebenen Term (Argument) ersetzt wird.

Formal lässt sich ein Ausdruck `(λx → s) t` äquivalent umformen zu `s[t/x]`, wenn diese Substitution erlaubt ist. Falls die Substitution nicht erlaubt ist, müssen zuvor noch α-Konversionen zum Umbenennen von Parametern verwendet werden.