

Inleiding Theoretische Informatica – Lambda Calculus

Uitwerkingen van geselecteerde opgaven

Martijn Vermaat (mvermaat@cs.vu.nl)

16 april 2007

1 Termen en reductie

2. Een functie die twee inputs neemt en ze afbeeldt op de constante 0 is $\lambda x.\lambda y.0$ of $\lambda xy.0$.

3. Met alle haakjes en alle λ 's.

(a) $((\lambda x.(\lambda y.(x(yz)))) (\lambda x.((yx)x)))$

4. Met zo min mogelijk haakjes.

(b) $(\lambda x.x) (\lambda x.\lambda y.x y y) \lambda x.\lambda y.x (x y)$

7. Zijn de termen α -convertibel?

(a) Nee.

(c) Nee.

(e) Nee.

(h) Ja.

8. Termen na uitvoeren van de substituties.

(b) $(\lambda x.\text{mul } x y)$

(c) $(\lambda x.\text{mul } x 5)$

(h) $(\lambda z.(\lambda y.\text{plus } x y) z)$

10. De termen worden gereduceerd naar normaalvorm en β -redexen zijn steeds onderstreept.

(a) $\underline{(\lambda y.\text{mul } 3 y) 7} \rightarrow_{\beta} \text{mul } 3 7 \rightarrow_{\delta} 21$

(d)

$$\begin{aligned} \underline{(\lambda f x.f x) (\lambda y.\text{plus } x y) 3} &\rightarrow_{\beta} \underline{(\lambda z.(\lambda y.\text{plus } x y) z) 3} \\ &\rightarrow_{\beta} \underline{(\lambda z.\text{plus } x z) 3} \\ &\rightarrow_{\beta} \text{plus } x 3 \end{aligned}$$

(f)

$$\begin{aligned}
(\lambda f x.f(f(x))) (\lambda x.\text{mul } x \ 7) \ 3 &\rightarrow_{\beta} (\lambda x.(\lambda x.\text{mul } x \ 7) ((\lambda x.\text{mul } x \ 7) (x))) \ 3 \\
&\rightarrow_{\beta} (\lambda x.(\lambda x.\text{mul } x \ 7) (\text{mul } x \ 7)) \ 3 \\
&\rightarrow_{\beta} (\lambda x.\text{mul } (\text{mul } x \ 7) \ 7) \ 3 \\
&\rightarrow_{\beta} \text{mul } (\text{mul } 3 \ 7) \ 7 \\
&\rightarrow_{\delta} \text{mul } 21 \ 7 \\
&\rightarrow_{\delta} 147
\end{aligned}$$

2 Reductiestrategieën

2. (a) Een reductierij volgens de leftmost-innermost strategie. Er is steeds maar één redex, dus is er geen verschil met het toepassen van bijvoorbeeld de leftmost-outermost strategie.

$$\begin{aligned}
(\lambda x.x \ x \ x) \ \lambda x.x \ x \ x &\rightarrow_{\beta} (\lambda x.x \ x \ x) \ (\lambda x.x \ x \ x) \ \lambda x.x \ x \ x \\
&\rightarrow_{\beta} (\lambda x.x \ x \ x) \ (\lambda x.x \ x \ x) \ (\lambda x.x \ x \ x) \ \lambda x.x \ x \ x \\
&\rightarrow_{\beta} (\lambda x.x \ x \ x) \ (\lambda x.x \ x \ x) \ (\lambda x.x \ x \ x) \ (\lambda x.x \ x \ x) \ \lambda x.x \ x \ x \\
&\rightarrow_{\beta} \dots \text{etc.}
\end{aligned}$$

- (c) De leftmost-outermost strategie vindt altijd een normaalvorm als deze bestaat. We vinden in dit geval geen normaalvorm, dus heeft deze term geen normaalvorm.

3 Datatypes

9. (a) Een specificatie voor de operatie *exclusive or*.

$$\begin{aligned}
\text{xor true true} &=_{\beta} \text{false} \\
\text{xor true false} &=_{\beta} \text{true} \\
\text{xor false true} &=_{\beta} \text{true} \\
\text{xor false false} &=_{\beta} \text{false}
\end{aligned}$$

- (b) Een mogelijke definitie voor de *exclusive or* operatie als λ -term. Volg de tip bij de vraag op voor een idee hoe te beginnen aan deze definitie.

$$\text{xor} := \lambda a b.b(a \ \text{false true}) \ a$$

- (c) Een afleiding voor de tweede regel van de specificatie bij (a). We vullen de definities van xor, true en false in en laten zien dat de term reduceert naar true.

$$\begin{aligned}
\text{xor true false} &= (\lambda a b.b(a \ \text{false true}) \ a) (\lambda xy.x) \ \lambda xy.y \\
&\rightarrow_{\beta} (\lambda b.b((\lambda xy.x) \ \text{false true}) (\lambda xy.x)) \ \lambda xy.y \\
&\rightarrow_{\beta} (\lambda xy.y) ((\lambda xy.x) \ \text{false true}) \ \lambda xy.x \\
&\rightarrow_{\beta} (\lambda y.y) \ \lambda xy.x \\
&\rightarrow_{\beta} \lambda xy.x \\
&= \text{true}
\end{aligned}$$

4 Recursie

7. We zijn op zoek naar een term `map` die het volgende gedrag vertoont:

$$\text{map} = \lambda fl. (\text{empty } l) \text{ nil } (\text{cons } (f \text{ (head } l)) (\text{map } f \text{ (tail } l)))$$

Dit kunnen we ook schrijven als

$$\text{map} = (\lambda m. \lambda fl. (\text{empty } l) \text{ nil } (\text{cons } (f \text{ (head } l)) (m \text{ } f \text{ (tail } l)))) \text{ map}$$

Nu kunnen we de fixed-point combinator Y gebruiken om de recursie op te lossen en een uiteindelijke definitie van `map` te maken:

$$\text{map} = Y \lambda m. \lambda fl. (\text{empty } l) \text{ nil } (\text{cons } (f \text{ (head } l)) (\text{map } f \text{ (tail } l)))$$

5 Getypeerde λ -calculus

1. (c) De term `KI` is gelijk aan $(\lambda xy.x) \lambda x.x$.

$$\frac{\frac{x : C \vdash x : C}{\vdash \lambda x.x : C \rightarrow C} \quad \frac{\frac{x : C \rightarrow C, y : B \vdash x : C \rightarrow C}{x : C \rightarrow C \vdash \lambda y.x : B \rightarrow C \rightarrow C}}{\vdash \lambda xy.x : (C \rightarrow C) \rightarrow B \rightarrow C \rightarrow C}}{\vdash (\lambda xy.x) \lambda x.x : B \rightarrow C \rightarrow C}$$