6. Übung - Programmierung

Kombinatoren und PROLOG

SS 18

Rekursion im λ-Kalkül

- Manche λ-Terme besitzen keine Normalform

  $f = (\lambda x.xx)(\lambda x.xx)$

  $\Rightarrow_\beta (\lambda x.xx)(\lambda x.xx)$

  ...

  $\Rightarrow_\beta^* (\lambda x.xx)(\lambda x.xx)$

Rekursion im λ-Kalkül

- Problem: Es gibt keine "globalen Variablen" $\Rightarrow$ Funktionen können sich nicht selbst aufrufen

- Lösung: Fixpunktkombinator

  Fixpunkt $\Rightarrow \langle f \rangle\ x = x$

  Y-Fixpunktkombinator $\Rightarrow \langle Y \rangle \langle f \rangle = \langle f \rangle (\langle Y \rangle \langle f \rangle)$

  - in javascript:

```
function Y(f){
    return f(Y(f))
}
```

  - in "Lambda":...

Y-Fixpunktkombinator

$$\langle Y \rangle = (\lambda h.((\lambda y.h(yy))((\lambda y.h(yy))))$$

$$\langle Y \rangle \langle F \rangle = (\lambda h.((\lambda y.h(yy))((\lambda y.h(yy)))) \ \langle F \rangle$$

$$\Rightarrow_\beta ((\lambda y.\langle F \rangle (yy)) \ ((\lambda y.\langle F \rangle (yy))) \qquad \text{entspr. } (\langle Y \rangle \langle F \rangle)$$

$$\Rightarrow_\beta \langle F \rangle \ (((\lambda y.\langle F \rangle (yy) \ (\lambda y.\langle F \rangle (yy)))) \qquad \text{entspr. } F \ (\langle Y \rangle \langle F \rangle)$$

also $\langle Y_F \rangle \Rightarrow^* \langle F \rangle \langle Y_F \rangle$

Aufgabe 1 b)

Nebenrechnung

$$\langle Y \rangle \langle F \rangle = (\lambda h.((\lambda y.h(yy))((\lambda y.h(yy)))) \ \langle F \rangle$$

$$\Rightarrow_\beta \underbrace{((\lambda y.\langle F \rangle (yy)) \ ((\lambda y.\langle F \rangle (yy)))}_{\langle Y_F \rangle}$$

$$\Rightarrow_\beta \langle F \rangle \langle Y_F \rangle$$

Daraus folgt $\langle Y_F \rangle \Rightarrow^* \langle F \rangle \langle Y_F \rangle$

Wiederholung - Kombinatoren

- Nomenklatur: $f = \lambda x.term \Rightarrow \langle \boldsymbol{f} \rangle$
- gängige Kombinatoren:

Symbol	Aufruf	Bedeutung
$\langle pred \rangle$,$\langle succ \rangle$	$\langle succ \rangle$ x	Vorgänger, Nachfolger
$\langle add \rangle$, $\langle sub \rangle$	$\langle add \rangle$ x y	Summe, Differenz
$\langle mul \rangle$	$\langle mul \rangle$ x y	Produkt
$\langle ite \rangle$	$\langle ite \rangle$ b t e	If b then t else e
$\langle iszero \rangle$	$\langle iszero \rangle$ x	Wahrheitswert von x==0

PROLOG

- **Syntax**:
  1. Variablen groß →X,
     Prädikate klein→ reif(X) , apfel
  2. Ende jeder Klausel → **.**
  3. und-Verknüpfung **,**
  4. oder-Verknüpfung **;**
  5. Regeloperator (Implikation) → **:-**
  6. Anfrage/Ziel → **?-**

PROLOG

Prinzip

- **Wissensdatenbank**:

  Fakten: obst(apfel).
  reif(apfel).

  Regeln: isGenießbar(X) :- obst(X), reif(X).

  $\widehat{=} \forall x.(istGenießbar(x) \leftarrow obst(x) \wedge reif(x))$

- **Anfrage-Verarbeitung**:

  ?- term1, term2, term3

  SLD-Resolution: Unifikation der Terme der Anfrage mit Fakten oder Implikationen der Wissenbasis
  $\rightarrow$ Ersetzen des Terms

  SLD-Refutation: Endliche Resolution mit Ziel: **?-.**

SLD- Refutation

Unfikator	
	?- prod(s(s(0)), s(0), X).
	?- prod(s(0), s(0), W), sum(s(0), W, X).
	?- prod(0, s(0), W1), sum(s(0), W1, W) , sum(s(0), W, X).
$\{W1 = 0\}$	?- prod(0, s(0), 0), sum(s(0), 0, W) , sum(s(0), W, X).
	?- sum(s(0), 0, W) , sum(s(0), W, X).
$\{W = s(Z1)\}$	?- sum(0, 0, Z1) , sum(s(0),s(Z1), X).
$\{Z1 = 0\}$	?- sum(s(0), s(0), X).
$\{X = s(Z2)\}$	?- sum(0,s(0), Z2).
$\{Z2 = s(0)\}$	?- .

„Computed Answer“ $X = s(Z2) = s(s(0))$