Programming languages - U6

Jan Dietrich - 10-100-436

1

Given:

$t \equiv \lambda f.\lambda n_1.\lambda n_2.$if (isZero n_1) 0 (add $n_2(f$(pred $n_1)n_2)$

and

$(((Yt)1)k) \to k$

we can replace the insert the values for n_1, n_2, f

if (isZero 1) 0 (add $k((Yt)$(pred $1)k)$

with isZero 1 = false und pred 1 = 0

add $k((Yt)0k$

since we have

$Yt \leftrightarrow t(Yt)$

we can reuse the function definition to achieve

add $k((\lambda f.\lambda n_1.\lambda n_2.$if (isZero n_1) 0 (add $n_2(f$(pred $n_1)n_2))0k$

with isZero n_1 with $n_1 = 0$ is true results in

add $k0$

since we know the implementation of add we can safely say

add k 0 = k

2

replace append with a f

app $\equiv \lambda f.\lambda l_1.\lambda l_2.$if (null $l_1)l_2$(cons (head $l_1)(f$(tail $l_1)l_2))$

To test insert L_1 = cons 1(cons 2 nil) and L_2 = cons 3 nil

if (null (cons 1(cons 2 nil))(cons 3 nil)(cons (head (cons 1(cons 2 nil)))(f(tail (cons 1(cons 2 nil)))(cons 3 nil))

since it's pretty unreadable we try to simplify

we can simplify the head and tail parts

if (null (cons 1(cons 2 nil))(cons 3 nil)(cons (cons 1)(f(cons 2 nil)(cons 3 nil))

next step we can simplify cons

if (null (cons 1(cons 2 nil))(cons 3 nil)(cons 1(f(cons 2 nil)(cons 3 nil))

we now that null (cons 1(cons 2 nil)) must be false

cons 1(f(cons 2 nil)(cons 3 nil))

and since Yapp $\leftrightarrow$ app(Yapp)

cons 1(app (Yapp)(cons 2 nil)(cons 3 nil))

By inserting the app definition

cons 1($\lambda f.\lambda l_1.\lambda l_2$.if (null l_1)l_2(cons (head l_1)(f(tail l_1)l_2))(Yapp)(cons 2 nil)(cons 3 nil))

with null cons 2 nil = false and head (cons 2 nil) = 2 and tail cons 2 nil = nil

cons 1(cons 2((Yapp)nil (cons 3 nil)

again use app definition with l_1 = nil

and since we now have

null nil which results to true

we can simpify to

cons 1(cons 2(cons 3 nil)

In the example there is written that this is equivalent to [1,2,3]