

Comp 205: Comparative Programming Languages

The Untyped λ -calculus

- representing values

Lecture notes, exercises, etc., can be found at:
www.csc.liv.ac.uk/~grant/Teaching/COMP205/

Computing with λ -Terms

How does the λ -calculus relate to real programming languages such as Miranda?

- λ -abstraction corresponds to defining functions with formal parameters, cf.:

$\lambda x. y = x$ $\lambda x. ?y. x$

- function application works in the same way, through substitution:

$\lambda x. M N \mapsto M$ $(\lambda x. ?y. x) M N \mapsto_{\beta} M$

Functions and Values

How can we represent **values** in the λ -calculus?

In Miranda we can write

$\text{twice } x = 2 * x$

but we can't (yet) write

$\lambda x. 2 * x$

since $2 * x$ isn't a λ -term.

Representing Booleans

Booleans can be represented in the λ -calculus as follows:

- True** is represented by $\lambda x. ?y. x$
- False** is represented by $\lambda x. ?y. y$

Case-Analysis

Given a λ -term B which is either **True** or **False**, a case-analysis

if B then M else N,

where M and N are arbitrary λ -terms, is represented by the λ -term $(B M) N$:

- if B is **True**: $(B M) N = ((\lambda x. ?y. x) M) N \mapsto_{\beta} M$
- if B is **False**: $(B M) N = ((\lambda x. ?y. y) M) N \mapsto_{\beta} N$

Representing Numbers

Natural numbers can be represented in the λ -calculus as follows:

- 0** is represented by $\lambda z. ?s. z$
- Succ 0** is represented by $\lambda z. ?s. s z$
- Succ (Succ 0)** is represented by $\lambda z. ?s. s (s z)$
- n** is represented by $\lambda z. ?s. s (s \dots (s z) \dots)$
(where s is applied n times to z)

Representing Constructors

A natural number n is represented by a λ -term N :

$\lambda z. \lambda s. s (s \dots (s z) \dots)$ (with s applied n times to z)

If we apply N to λ -terms Z and S , the effect is to replace z with Z and each s with S :

$N Z S \equiv_{\beta} S (S \dots (S Z) \dots)$

Applying S to each side gives

$S (N Z S) \equiv_{\beta} S (S (S \dots (S Z) \dots))$ (with $n+1$ S 's)

...

Representing Constructors

$S (N Z S) \equiv_{\beta} S (S (S \dots (S Z) \dots))$ (with $n+1$ S 's)

Therefore

$\lambda z. \lambda s. s (N z s)$

represents $n+1$,
and **Succ** is represented by the λ -term

$\lambda n. \lambda z. \lambda s. s (n z s)$

Working with Numbers

Given M and N :

$M Z S \equiv_{\beta} S (S \dots (S Z) \dots)$ (with m S 's)
 $N Z S \equiv_{\beta} S (S \dots (S Z) \dots)$ (with n S 's)

If we replace Z in the former by $N Z S$, we get

$M (N Z S) S \equiv_{\beta} S (S \dots (S (S \dots (S Z) \dots))$ (with $n+m$ S 's)

Therefore $M+N$ is represented by $\lambda z. \lambda s. M (N z s) s$
and addition is the λ -term

$\lambda m. \lambda n. \lambda z. \lambda s. m (n z s) s$

Summary

Key points:

- representing booleans
- representing numbers
- representing operations on numbers

Next: Evaluation Strategies and Types