

MATH230: Tutorial Seven
Recursion and Combinatory Logic

Key ideas

- Write recursive processes in λ -calculus,
- Write higher order procedures in λ -calculus,
- Prove extensional identities in combinatory logic,
- Translate between λ -calculus and combinatory logic.

Relevant topic: Untyped Lambda Calculus Slides

Relevant reading: Type Theory and Functional Programming, Simon Thompson

Hand in exercises: 1b, 4c, 5c, 6a, 7c

Due Friday @ 5pm to the submission box on Learn.

m

Discussion Questions

- Determine some steps towards writing a program (λ -term) representing the unary function, INT-SQRT, that returns the greatest natural number whose square is less than or equal to the input.

Tutorial Exercises

1. Write recursive λ -expressions that represent the following functions of natural numbers. For each function determine an appropriate helper-function GO to put through the Y combinator.
 - (a) SUM of two natural numbers
 - (b) MULTiply two natural numbers
 - (c) EXPONentiation of a base to an exponent
 - (d) FACTorial of a natural number
 - (e) INT-SQRT the smallest integer whose square is greater than input
 - (f) Calculate the nth FIBonacci number (Challenge!)
2. Write a λ -expression that can be used to compute the smallest natural number that satisfies a given unary-predicate $P?(x)$ that is represented by some λ -expression.
3. (Challenge!) Represent the following processes in the λ -calculus to get an expression that can be used to test whether a natural number is prime. For simplicity, assume the input is greater than TWO.
 - (a) REMAINDER calculate the remainder of a division.
 - (b) DIVIDES? binary predicate does second divide first?
 - (c) Implement bounded-search to satisfy a predicate.
 - (d) PRIME? Unary-predicate to detect primality.
4. In lectures we introduced a λ -term for computing the sum of a sequence of consecutive integers. This used the helper-function:

$$\text{GO} \equiv \lambda s. \lambda a. \lambda l. \lambda u. \text{COND } (>? \ l \ u) \\ a \\ (s \ (\text{SUM } a \ l) \ (\text{SUCC } l) \ u)$$

We defined $\text{ACCUMULATE} = \text{Y GO}$. Make alterations to the helper-function to compute the following:

- (a) Compute the sum of the squares of each integer, $\sum_{i=l}^u i^2$
- (b) Compute the sum of each term passed through an arbitrary function, $\sum_{i=l}^u f(i)$
- (c) Compute the sum of those terms in the interval that satisfy some predicate $P?(x)$.

Recall the following reduction rules of the CL combinators.

$$\begin{array}{ll} \mathbf{S}xyz \rightarrow_{\beta} xz(yz) & \mathbf{K}xy \rightarrow_{\beta} x \\ \mathbf{I}x \rightarrow_{\beta} x & \mathbf{B}fgx \rightarrow_{\beta} f(gx) \\ \mathbf{W}fx \rightarrow_{\beta} fxx & \mathbf{C}fxy \rightarrow_{\beta} fyx \end{array}$$

5. Verify each of the following extensional equality claims by evaluating each side at an appropriate number of variables and check the reductions are identical.

- (a) $\mathbf{I} = \mathbf{SKK}$
- (b) $\mathbf{SK} = \mathbf{KI}$
- (c) $\mathbf{B} = \mathbf{S(KS)K}$
- (d) $\mathbf{W} = \mathbf{SS(KI)}$
- (e) $\mathbf{C} = \mathbf{S(BBS)(KK)}$

6. Each of these CL terms are reducible. Compute their normal form.

- (a) $\mathbf{SKI(KIS)}$
- (b) $\mathbf{KS(I(SKSI))}$
- (c) \mathbf{SKIK}

7. Translate each of the λ -terms into combinatory logic expressions involving only **SKI** combinators using the translation defined in the slides.

- (a) $\lambda x. \lambda y. y$
- (b) $\lambda c. \lambda f. \lambda g. c f g$
- (c) $\lambda s. \lambda x. s x$
- (d) $\lambda f. (\lambda x. f x x) (\lambda x. f x x)$