

Categorifying the λ -calculus

Last time we saw that a presentation of some set-based algebraic structure, e.g. a monoid, also gives rise to a category if we interpret the relations not as equations but rewrite rules. We get a category with:

- objects being expressions built from the generators using the operations in our structure.
- morphisms being built from the rewrite rules using composition and the operations in our structure.

Now we'll do something similar where our algebraic structure will be "a typed λ -calculus" as defined by Lambek & Scott (without their "natural numbers object"). Lambek & Scott show that a typed λ -calculus can serve as a presentation of a cartesian closed category (CCC). A CCC is already a category-based algebraic structure, so if we treat Lambek & Scott's relations as rewrite rules, we'll get a 2-categorical structure: hopefully a "cartesian closed 2-category." In such a thing, we should have:

- objects being types A, B, C, \dots
 including product types $A \times B, \dots$
 function types $\text{hom}(A, B), \dots$

- morphisms $f: A \rightarrow B$ are pairs $(x \in A, \varphi(x))$ where $\varphi(x)$ is a λ -term of type B including the variable x as a free variable.

e.g.
$$\varphi(x) = \underbrace{y \in \text{hom}(A, C) \mapsto y(x)}_{\substack{\text{a } \lambda\text{-term of type} \\ \text{hom}(\text{hom}(A, C), C)}}$$

So for each $x \in A$, the pair $(x, \varphi(x))$ is a morphism

$$f: A \rightarrow \text{hom}(\text{hom}(A, C), C)$$

- 2-morphisms are built from rewrite rules such as " β -reduction", " η -reduction".

More precisely, a typed λ -calculus is

- 1) A set of types s.t.
 - a) There's a type 1 .
 - b) Given types A and B , there's the product type $A \times B$ & function type $\text{hom}(A, B)$

3) Relations between terms. Every relation is of the form

$$a \stackrel{X}{=} a'$$

where a, a' are terms of type A & X is a ^{finite set} list of variables which includes all the free variables in a or a' .

(A variable x ceases to be free - it becomes bound - when it appears in an expression $x \in A \mapsto \varphi(x)$. I.e. here x is a dummy variable.)

a) $\stackrel{X}{=}$ is an equivalence relation (in Lambek & Scott's definition - we'll probably drop the symmetry!)

$$a \stackrel{X}{=} a$$

$$a \stackrel{X}{=} a' \text{ \& \& } a' \stackrel{X}{=} a'' \implies a \stackrel{X}{=} a''$$

$$a \stackrel{X}{=} a' \implies a' \stackrel{X}{=} a$$

b) If $X \subseteq X'$ then

$$a \stackrel{X}{=} a' \implies a \stackrel{X'}{=} a'$$

c) If $x \in A$ & $\varphi(x) \stackrel{X \cup \{x\}}{=} \varphi'(x)$, then

$$(x \in A \mapsto \varphi(x)) \stackrel{X}{=} (x \in A \mapsto \varphi'(x))$$

d) $a \stackrel{X}{=} a' \implies f(a) \stackrel{X}{=} f(a')$

$$e) a \stackrel{\equiv}{X} * \quad \forall a \in 1$$

$$f) \pi_1(a,b) \stackrel{\equiv}{X} a \quad \pi_2(a,b) \stackrel{\equiv}{X} b$$

$$(\pi_1(c), \pi_2(c)) \stackrel{\equiv}{X} c \quad \text{for all } c \in A \times B$$

$$g) (x \in A \mapsto \varphi(x))(a) \stackrel{\equiv}{X} \varphi(a)$$

for all $a \in A$ s.t. no free variable in a becomes bound in $\varphi(a)$

$$h) x \in A \mapsto f(x) \stackrel{\equiv}{X} f$$

if $f \in \text{hom}(A, B)$ & $x \in X$ (i.e. a variable in the set X).

$$i) x \mapsto \varphi(x) \stackrel{\equiv}{X} y \mapsto \varphi(y)$$

if no free occurrence of x in $x \mapsto \varphi(x)$ becomes bound in $y \mapsto \varphi(y)$.