CLASSICAL vs. QUANTUM COMPUTATION

Winter 2007

4 Jan 2007

Last time we discussed how computation is related to monoidal closed categories — categories with $\otimes$ and internal hom. The big idea is:

CATEGORIES	COMPUTATION
Objects	Data types
Morphisms $f: A \to B$	Equivalence classes of programs that accept data of type A and output data of type B

Example: if our category is Set, a typical object might be $\mathbb{N}$ and a typical morphism might be

$$+ : \mathbb{N} \times \mathbb{N} \longrightarrow \mathbb{N}$$

where $\times$ is the tensor product in Set. Note that lots of different programs compute the fn $+ : \mathbb{N} \times \mathbb{N} \longrightarrow \mathbb{N}$, so our morphisms are equivalence classes of programs.

The problem with using equivalence classes is that very different programs can be identified. We've seen that in any monoidal closed category we have:

$$(x \longmapsto f(x))(y) = f(y) \qquad \text{"}\beta\text{-reduction"}$$

where "=" really is our equivalence relation.

Similarly: $f = (x \longmapsto f(x))$ "η-expansion"

These equivalence relations ignore the process of doing computation by saying, for example, that a program that computes the answer to something and gets 8 is equal to a program that just prints out the number 8. The problem is: we're treating isomorphic things as equal and ignoring the isomorphism. There's a general strategy for solving this kind of problem:

CATEGORIFICATION

In categorification, we convert set-based math into category based math as follows:

Set based math	Category
elements	objects
equations between elements	(iso)morphisms between objects
sets	categories
functions btwn. sets	functors btwn. categories
equations btwn. functions	natural transformations (or isomorphisms) btwn. functors

Categorification promotes equations to (natural) isomorphisms.

Example: A monoid is a set S with a multiplication function

$$\cdot : S \times S \longrightarrow S$$

and an identity element

$$1 \in S$$

satisfying equations:

$$(a \cdot b) \cdot c = a \cdot (b \cdot c)$$
$$1 \cdot a = a \cdot 1$$

This is set-based math. Categorifying, we get:

A monoidal category is a category C with a multiplication functor

$$\otimes : C \times C \longrightarrow C$$

and an identity object

$$1 \in C$$

equipped with isomorphisms

$$\alpha_{a,b,c} : (a \otimes b) \otimes c \xrightarrow{\sim} a \otimes (b \otimes c)$$
$$1 \otimes a \xrightarrow[\sim]{l_a} a \xleftarrow[\sim]{r_a} b \otimes 1$$

satisfying equations of their own (this is the hard part!). Namely, the pentagon identity for the associator & some equations for left & right unitors.

An example of a monoid: $(\mathbb{N}, +, 0)$, the free monoid on 1. This is related to a certain monoidal category, $(\mathrm{FinSet}, +, \emptyset)$ where $+$ is product. If we decategorify FinSet - i.e. form the set of isomorphism classes of objects - you get the set $\mathbb{N}$. Decategorifying any monoidal category you get a monoid. In fact, FinSet is the free category with finite coproducts on one object.

In our problem, we're working with equivalence classes of programs as morphisms — instead of treating them as equal, we should treat them as isomorphic. To do this — to talk about isomorphic morphisms — we need to categorify the concept of category itself!

A category is a collection of objects & for any pair of objects A, B, a set of morphisms $\mathrm{Hom}(A,B)$, any triple of objects A, B, C a composition function

$$\circ : \mathrm{Hom}(A,B) \times \mathrm{Hom}(B,C) \longrightarrow \mathrm{Hom}(A,C)$$

and for an object A an element

$$1_A \in \mathrm{Hom}(A,A)$$

satisfying some equations — associativity and l/r unit laws. (This is a lot like the definition of monoid, since a monoid is just a category with one object.)

If we make the replacements:

set $\longmapsto$ category

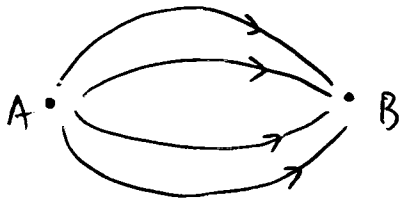
function $\longmapsto$ functor

element $\longmapsto$ object

equations $\longmapsto$ natural isomorphisms

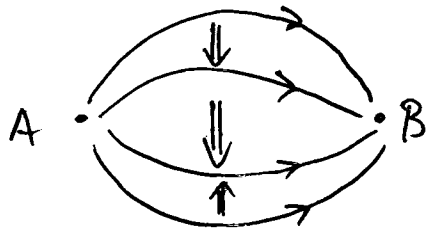
and again impose the pentagon identity & identity for l/r unitors, we get the definition of (weak) 2-category or bicategory

Category



$\mathrm{Hom}(A,B)$ is a set

2-category



$\mathrm{Hom}(A,B)$ is a category

Now we have 2-morphisms between morphisms, so we can talk about isomorphic morphisms.