17 May 2007

The Bar Construction

Why do adjoint functors

$$
\begin{tikzcd}
D \arrow[d, shift left, "R"] \\
C \arrow[u, shift left, "L"]
\end{tikzcd}
$$

"free" L, R "forgetful"

give simplicial objects

$$
\begin{tikzcd}
d & LRd \arrow[l, bend right, "\varepsilon_d"'] \arrow[r, bend right, "L(\iota_{Rd})"'] & LRLRd \arrow[l, bend right, "\varepsilon_{LRd}"'] \arrow[l, "LR(\varepsilon_d)"'] \arrow[r, bend right] \arrow[r, bend right=50] & \cdots \arrow[l, bend right] \arrow[l, bend right=40] \arrow[l, bend right=60]
\end{tikzcd}
$$

in D from objects $d \in D$? In other words, why do we get a functor

$$\bar{d} : \Delta^{op} \longrightarrow D$$

(the bar construction applied to $d \in D$)?

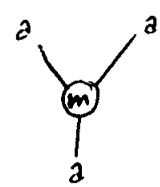
To answer these questions, we need to learn more about Δ & adjoint functors — which are closely connected!

First,

"Δ is the walking monoid".

In other words, Δ's sole purpose in life is to contain a monoid. A monoid, naively speaking, is a set with an associative product and unit. This is just a monoid in Set; more generally, a monoid can be defined in any monoidal category. Given a monoidal category C, a monoid in C is an object $a \in C$ together with a product:

$$m : a \otimes a \longrightarrow a$$



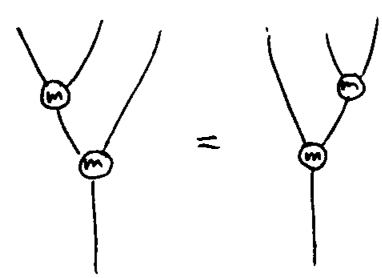
and unit

$$i : 1 \longrightarrow a$$

satisfying the associative law

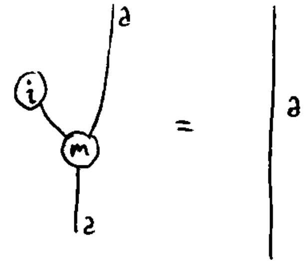
$$\begin{tikzcd} & a \otimes a \otimes a \arrow[dl, "m \otimes 1"'] \arrow[dr, "1 \otimes m"] & \\ a \otimes a \arrow[dr, "m"'] & & a \otimes a \arrow[dl, "m"] \\ & a & \end{tikzcd}$$

commutes.

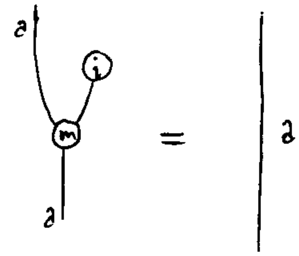


(where for expository purposes, we assume C is strict, so $(a \otimes b) \otimes c = a \otimes (b \otimes c)$) and the left/right unit laws; saying these diagrams commute:

$$\begin{tikzcd} a \arrow[r, equal] \arrow[d, "1_a"'] & 1 \otimes a \arrow[d, "i \otimes 1_a"] \\ a & a \otimes a \arrow[l, "m"] \end{tikzcd}$$



$$\begin{tikzcd} a \otimes 1 \arrow[r, equal] \arrow[d, "1_a \otimes i"'] & a \arrow[d, "1_a"] \\ a \otimes a \arrow[r, "m"'] & a \end{tikzcd}$$



How is Δ a monoidal category, and what's the monoid in it? Recall Δ has

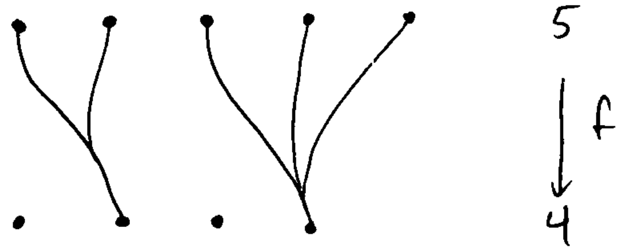
- finite totally ordered sets as objects
- order-preserving maps as morphisms.

We write the finite totally ordered sets as finite ordinals $0, 1, 2, 3, \ldots$. We can make Δ into a monoidal category with $+$ as tensor product:

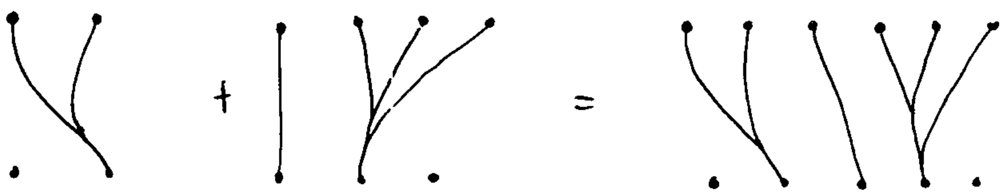
totally ordered.

$$\underset{3}{\bullet \; \bullet \; \bullet} \; + \; \underset{2}{\bullet \; \bullet} \; = \; \underset{5}{\bullet \; \bullet \; \bullet \; \bullet \; \bullet}$$

on objects, but also on morphisms. A typical morphism in Δ looks like



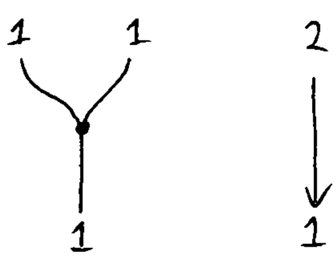
(f must be order-preserving), and we tensor ($+$) them as follows



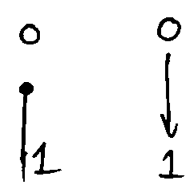
So Δ is a monoidal category; what's the obvious monoid in it? It's some totally ordered finite set a with order preserving maps

$$m : a + a \longrightarrow a$$
$$i : 0 \longrightarrow a$$

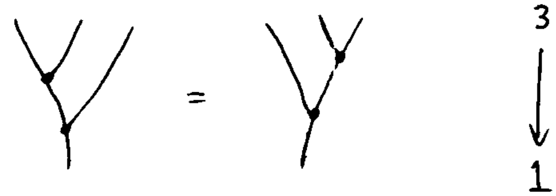
satisfying assoc. & l/r unit laws. We take $a = 1$, take $m : a + a \to a$ to be the only thing it could be



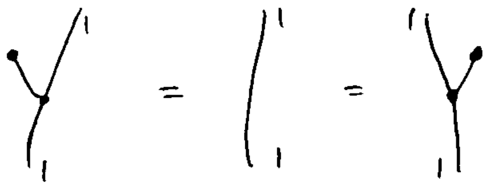
& take $i : 0 \to 1$ to be the only thing it could be:



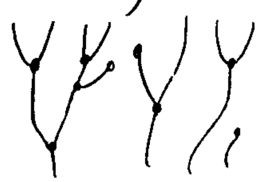
These satisfy associativity:



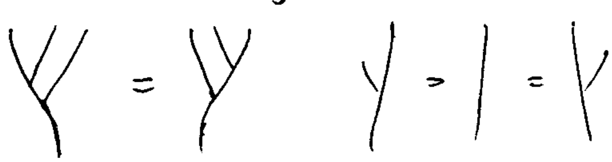
and l/r unit laws:



So Δ is a monoidal category containing a monoid; in fact it's the free monoidal category on a monoid. If we freely generate a monoidal category from a monoid we get a category with morphisms like



satisfying relations including



& monoidal category axioms. This is just Δ!

Second:

adjoint functors give monoids.

Given some adjoint functors

$$\begin{tikzcd} D \arrow[d, shift left, "R"] \\ C \arrow[u, shift left, "L"] \end{tikzcd}$$

we have the unit

$$\iota_c : c \longrightarrow RLc \qquad \text{(think "inclusion of the generators")}$$

and counit

$$\varepsilon_d : LRd \longrightarrow d \qquad \text{(think "evaluation of formal expressions")}$$

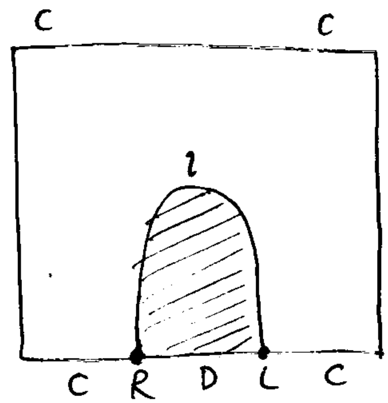
Note we have:

- categories C, D
- functors L, R
- natural trans. ι, ε

So we're talking about a situation in Cat, the 2-category of categories. We can draw what's going on using string diagrams, which apply to 2-categories as well as monoidal categories. We draw the unit

$$\iota : 1_C \Longrightarrow RL$$

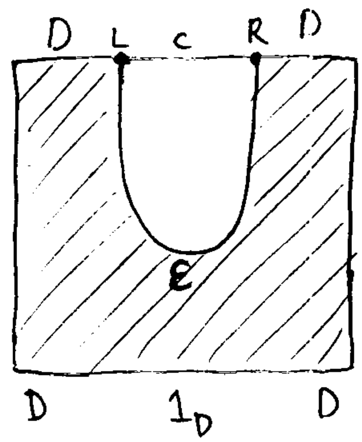
as



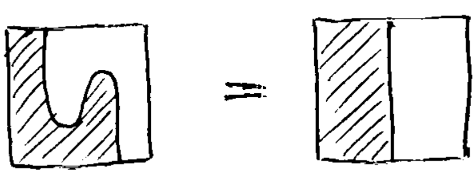
(We shade the "D" region, leaving the "C" region white.)

Similarly, we draw the counit $\varepsilon : LR \Longrightarrow 1_D$

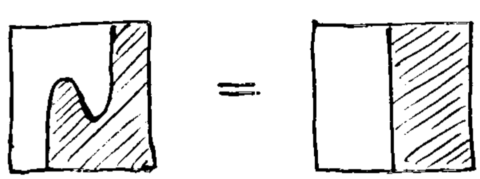
as:



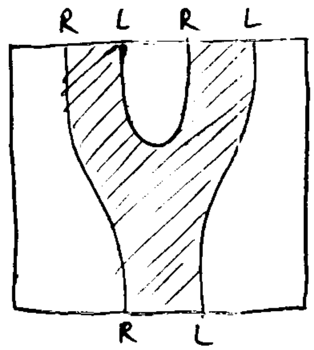
In fact, whenever we have adjoint functors, the zig-zag equations hold:



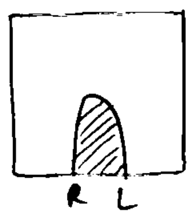
&



We'll get a monoid with product



& unit:



This kind of monoid is called a monad.